FACULTY OF SCIENCES

Giving physical constraints to supernovae from GW observations

Master thesis

Promotors: Dr. Gergely Dálya
Prof. Dr. Archisman Ghosh

Student: Sibe Bleuzé sibe.bleuze@ugent.be

Academic year: 2021–2022

GHENT UNIVERSITY

Acknowledgements

First of all, I would like to thank Dr. Gergely Dálya for introducing me to the world of supernova-induced gravitational waves and for his enormous patience in explaining things over and again if I'd forgotten how they worked. Without him I couldn't have delivered what you're about to read.
Of course, the same goes for Prof. Dr. Archisman Ghosh, whose enthousiasm was contagious. He's the one who sparked my interest in gravitational wave physics and helped me pick this fantastic topic in the first place.
I cannot forget to mention Prof. Rafael da Silva de Souza, as he helped me get started on the machine learning algorithms when I didn't yet fully understand how they all worked.

A special thanks goes out to my family and anyone else who proofread (part of) my thesis. Although I'm not sure how much of my explanations they understood, they always kept asking me about what I was doing and how things were going. This provided the necessary pressure for me to keep working, even if sometimes I didn't feel up for it.
My friends here at the university also deserve a word of thanks for their support. I may not always have been the most pleasant company, but I could always vent with them, which helped me put things into perspective.
My last words here are reserved for the student organization WiNA. They provided the breaks I didn't always realize I needed.

Abstract

On February 24th, 1987, a supernova explosion in the Large Magellanic Cloud was observed. At a distance of approximately $50\ \mathrm{kpc}$, it is the most recent close-by core-collapse supernova observed. It is also the only one to date that has been observed not only in optical observations, but also by a neutrino observatory (Hirata et al., 1987). This event, better known as SN1987A, has since provided the world with a wealth of knowledge about core-collapse supernova explosions and their connection to neutrino physics.

Since then, the observatories have improved and as of a few years ago, another entirely new field of observation has come within reach: gravitational waves. While they do not consist of two bodies like the binary black hole mergers that have already been observed, core-collapse supernovae are generally thought to be capable of producing gravitational radiation. This means that the next galactic supernova will potentially again be a huge source of information, at least if we are prepared to process the detection. That is why there is already a lot of research into what the gravitational emission of core-collapse supernovae would look like and how far away a supernova may be before it is outside of the detection range.

This thesis investigates, given that the next galactic core-collapse supernova would be detectable, what could be learned from its gravitational radiation alone. Areas of interest are its physical properties, like for example the mass of the stellar progenitor, and its underlying characteristics, like the explosion mechanism. For this purpose, a machine learning algorithm is trained and tested on the expected observational data from a set of simulated gravitational waves. The performance evaluation on the test data serves as a measure of how precise the value of a property or the presence of a characteristic can be determined.

Table of Contents

List of Figures

List of Acronyms

AGB asymptotic giant branch

BH black hole

CCSN core-collapse supernova

DT decision tree

EGO European Gravitational Observatory

EoS equation of state

ET Einstein Telescope

GW gravitational wave

HB horizontal branch

HRD Hertzsprung-Russell diagram

KAGRA Kamioka Gravitational Wave Detector

KNN K-nearest neighbour

LASSO least absolute shrinkage and selection operator

LIGO Laser Interferometer Gravitational-Wave Observatory

LISA Laser Interferometer Space Antenna

LR linear regression

LS Lattimer-Swesty

MHD magnetohydrodynamics

ML machine learning

MS main sequence

NaN not a number

NN neural network

NRMSE normalized root mean squared error

O4 observing run 4

O5 observing run 5

PCA principal component analysis

PNS proto-neutron star

RGB red giant branch

RJMCMC Reversible Jump Markov Chain Monte Carlo

RMF relativistic mean field

SASI standing accretion shock instability

SGD stochastic gradient descent

SN supernova

SNR signal to noise ratio

SVM support vector machine

TM Thomas-Fermi

UMAP uniform manifold approximation and projection

XOR exclusive or

Chapter 1

Introduction

1.1 Supernovae

1.1.1 Stellar evolution

The life cycle of a star is a much studied subject. Starting on the main sequence (MS), a star traverses a path of increase and decrease in brightness as it goes through different phases of nuclear burning. This path is usually captured on a Hertzsprung-Russell diagram (HRD), which plots the luminosity of a star as a function of its temperature. Highly populated areas of this diagram have their own names, such as the red giant branch (RGB), the horizontal branch (HB) and the asymptotic giant branch (AGB). Stars often eject mass during their life cycle, as a result of which it is a bit ambiguous to talk about *the* mass of a star. That is why in most contexts, the mass of a star refers to its mass at the start of its life on the main sequence, before any significant amount of mass has been ejected.

The path of each star is unique and depends on a number of factors, the most important of which is its main sequence mass. Most stars will go through the phases associated with the previously mentioned highly populated branches, but the end of their life cycle may differ quite a bit. The heaviest stars will reach a point where they collapse under their own gravity and form black holes (BHs). The black holes formed in this way are classified as stellar black holes and their masses range from $5\ \mathrm{M}_\odot$ up to $100\ \mathrm{M}_\odot$ after black hole formation. The lightest stars, up to approximately $8\ \mathrm{M}_\odot$, lose enough mass during their life cycle so that in the end, when nuclear burning stops, the electron degeneracy pressure of the stellar core can support the remaining mass. The maximum remaining mass for which this is possible is known as the Chandrasekhar limit, which lies around $1.44\ \mathrm{M}_\odot$.

In between these two mass ranges is a category of medium mass stars that, contrary to their lighter and heavier counterparts, literally go out with a bang. They exceed the Chandrasekhar limit at the end of their lives, meaning the electron degeneracy pressure cannot prevent their collapse. While collapsing into increasingly higher densities, they reach a point where even the

nucleons start to become degenerate. As the nucleon degeneracy pressure is much higher than that of the electrons, the material of the collapsing core bounces off itself, creating a shock wave outwards into the infalling material from the outer shells. When this shock wave reaches the surface of the star, the supernova explosion is complete.

While the overall mechanism is the same for all core-collapse supernovae (CCSNe), they are usually separated into subcategories based on their emission spectra. For the purpose of this thesis, only one distinction is of importance, which is the distinction between supernovae (SNe) of type Ia, which have a slightly different mechanism than the one described above, and core-collapse supernovae, regardless of their emission spectra. A type Ia supernova occurs when a white dwarf in a binary system grows in mass by accretion from its binary counterpart. When this causes it to exceed the Chandrasekhar limit, a supernova explosion is set in motion as if it were a more massive star all along. Since their mass at the point of explosion is always the same, so are their light curves. This makes type Ia supernovae very useful in astronomy as standard candles. However, even if they are asymmetric enough to produce gravitational waves (GWs), these would be the faintest ones possible from a supernova explosion. That is why this thesis will focus on core-collapse supernovae. They originate from heavier progenitors[1], they do not require a binary system and they can potentially produce a detectable GW signal.

Further reading about stars and their evolution on a BSc physics and astronomy level can be done in Dálya (2021).

1.1.2 Core-collapse supernovae

While the mechanism of CCSNe described in the previous section might sound simple enough, there are still a lot of challenges in explaining the physical details of a supernova explosion. Simulations show a number of phenomena occurring under the stellar surface during the imploding and exploding phases, like sloshing movements, convection and oscillatory behaviour. There are also multiple proposals for a mechanism to push further the shock front, which has been shown not to be able to sustain itself. The initial shock wave after core bounce stops after a few tens of milliseconds and needs another process to restart it. A discussion of the proposed mechanisms can be found further on in section 1.1.4.

One particular difficulty in studying supernovae is their distance. While the occurence of supernovae anywhere in the universe may be quite high, most of them occur in distant galaxies and can only be seen through the light they emit. It is only for close-by occurences that the neutrino and gravitational radiation can lead to a useful detection. For state-of-the-art neutrino detectors like the proposed Hyper-K, 'close-by' would extend to a few Mpc (Migenda, 2017). For GWs, even that is too far and the limits lie around a few tens or at best a few hundreds of kpc (Szczepańczyk et al., 2021). That is why the astrophysics community is eagerly awaiting the next galactic supernova. They may however need quite some patience, as supernovae inside our galaxy are quite rare. According to recent predictions, they only occur once or twice per century (Rozwadowska et al., 2021).

[1] pre-collapse stars, also referred to as proto-neutron star (PNS)

Listed below are a few characteristics of and proposed phenomena occurring during supernova explosions and the implications on physics if they would be present in observations.

1.1.3 Equation of state

The equation of state (EoS) relates several properties of a gas, like pressure, temperature and density. While the most basic equation of state, the ideal gas law, may be well known and simple to understand, the same cannot be said about the equation of state of supernovae. As the gas consists of nuclear matter rather than atoms in molecules, factors like the proton-neutron asymmetry or iso-spin dependence become of importance. This immediately links the study of supernovae to high energy nuclear physics, since these kinds of asymmetries in the proton-neutron balance can only be achieved on earth in rare isotopes resulting from accelerator collisions.

As with many physical problems, finding the exact solution for the supernova equation of state will probably never happen. Rather the existing numerical solutions will be able to improve upon themselves as observations confirm or exclude a certain parameter range for the variables that need to be fit. As no sufficient amount of observations exists yet, there are currently quite a lot of different equations of state used in different models. Some recurring EoSs are the Lattimer-Swesty (LS) equation of state, which uses a compressible liquid drop model, EoSs based on the Thomas-Fermi (TM) model and variational approximations with a relativistic mean field (RMF) model, as well as the more recently added SFH models (Steiner et al., 2013). Equations of state are usually presented in EoS tables after being calculated using a parameter grid for the temperature, density and composition, as was done for example in Shen et al. (2011).

When it comes to comparing equations of state, apart from looking at the exact parameter values, there's not much to go on. One characteristic that can be used as a comparison mechanism is the softness or stiffness. A stiff equation of state is one where pressure rises sharply with density. If a star would exhibit such an EoS, it would not be easily compressible and hence be more resistant to implosion under gravity. This of course would influence a supernova explosion as that requires an implosion to begin with. A soft equation of state only has a small response in pressure for a given change in density, hence the corresponding star would be more easily compressed.

1.1.4 Explosion mechanism

The first explosion mechanism, or more accurately shock revival mechanism, was proposed as early as 1985 by Bethe and Wilson (1985). It depends on the realization that even though neutrinos have an incredibly small cross section, the density inside the collapsing star is so high that even they cannot escape its interior freely. Instead, like the rest of the material moving outward after bouncing off the core, they accumulate at the stalled shock front and part of their energy is transferred to the nuclear material. It is this deposition of energy that ultimately revives the shock wave and allows it to reach the surface of the exploding star.

The mechanism described above is called neutrino heating, the corresponding supernova is said to exhibit a neutrino-driven explosion. Less than two years after its proposition, the core-collapse supernova SN1987A was observed, for the first time adding a neutrino observation to the usual optical signal. While at the time only 25 neutrino events were observed in three detectors combined, a future supernova would certainly be very clear, producing possibly thousands of events in the current detectors. Adding gravitational wave (GW) observations to such a detection would be the icing on the cake.

Of course, neutrino heating is not the only thing people have come up with over the years. A second mechanism that gained some popularity is the magneto-rotational or magnetohydrodynamics (MHD) explosion mechanism. Like the previous mechanism, it was proposed quite early on, this time by Bisnovatyi-Kogan (1971). With a strong magnetic field and a rotating stellar progenitor, it is possible to revive the shock wave by the transfer of momentum from the core outward to the stalled shock wave, pushing the outgoing material towards the surface. Unfortunately, not many successful simulations exist yet that show this explosion mechanism at work, so only two of them can be included in this thesis.

A third mechanism is acoustic in nature. It involves, among other things, the g-modes described in section 1.1.7. Burrows et al. (2006) found that when neither neutrino heating nor magnetorotational effects become dominant too early in the simulation, sound waves can build up an acoustic power large enough to drive the supernova explosion.

1.1.5 Standing accretion shock instability

While carrying out idealized 2D simulations of an accretion shock, Blondin et al. (2003) found that a standing accretion shock in an axisymmetric environment would become unstable under radial perturbations. The standing accretion shock instability (SASI) emerges as asymmetric sound waves characterized by low-order spherical harmonics ($l = 1, 2$). These waves grow until they eventually begin to impact the shock wave itself.

One effect of the SASI is the expansion of the average shock radius. Its sloshing-type motion thus generally pushes material outwards, lending a helping hand in the fight against the stalled shock front. This makes it potentially crucial in the explosion mechanism, which is why it is one of the reasons its signature will be searched for in the gravitational wave observations. A second reason is its potential connection to the equation of state, which will be explained in section 2.1.5. While it was discovered in 2D simulations, a lot of 3D simulations also show SASI activity. The difference is that in 3D it has more complex, nonaxisymmetric modes and has been observed to redistribute angular momentum.

1.1.6 Prompt convection

When the supernova shock wave stalls, there are quite strong gradients left in its path, since it blew almost everything away in front of itself. An example is the negative lepton gradient, caused by the increased interaction of electron neutrinos resulting in a drop in their density.

When gradients are concerned, convection is never far away, and prompt convection is the result of physics trying to smooth out the contrast between the regions in front of and behind the shock wave. This in itself may then cause even more energy-producing interactions, thus creating another way of refueling the shock wave. While its exact impact may be still up for discussion, prompt convection itself has been observed in simulations, so determining its presence in observations would certainly matter in establishing the different contributions to shock revival.

1.1.7 g-modes

Apart from the SASI, other oscillatory behaviour may occur in a supernova explosion. One of the most common type of oscillations is known as g-modes. They are a global oscillation with a number of nodes that is not predefined. Because gravity is the dominant force in restoring them, they are called gravity modes, or shorter g-modes. As they involve large-scale movements of mass, they will also produce a signature in the gravitational waves emitted from a supernova. However, since they are so common, they are present in all the simulations included in this thesis. Hence it will not be investigated if the machine learning algorithm can distinguish them, as there are no counterexamples present.

1.2 Gravitational waves

Supernovae, while fascinating events on their own, are not new to humanity. While the first observations were seen only with the naked eye, recorded events might go back thousands of years[2] (Hamacher, 2014). The increased interest in supernovae in the past few decades has been on par with advances in observational techniques, which can now be extended to not only include the visible light, but also consist of neutrino observations and of course, as of a few years ago, gravitational waves. In order to understand what can be learned about supernovae from gravitational wave observations, it is important to first get a good understanding of what gravitational waves are and how supernovae can produce them.

1.2.1 Measurement

Gravitational waves are the result of mass that is accelerated in an asymmetric way. An example of this consists of two objects that are orbiting each other. While in theory these bodies can have any mass, they will need to have quite a significant mass, of the order of a few solar masses, in order to be detectable, so things like exoplanets orbiting their star will not suffice[3]. Describing a gravitational wave can be done using the same properties used for light waves. They are, however, measured by completely different instruments and as a result of that, they are usually characterised differently as well. Where light already has a broad

[2]The uncertainty originates in the dating of the records in question as well as the uncertainty if said records pertain to a supernova or another unrelated event in the sky.

[3]Not yet at least, maybe the heaviest ones will in the future, but as exciting as that may sound, it is off topic here.

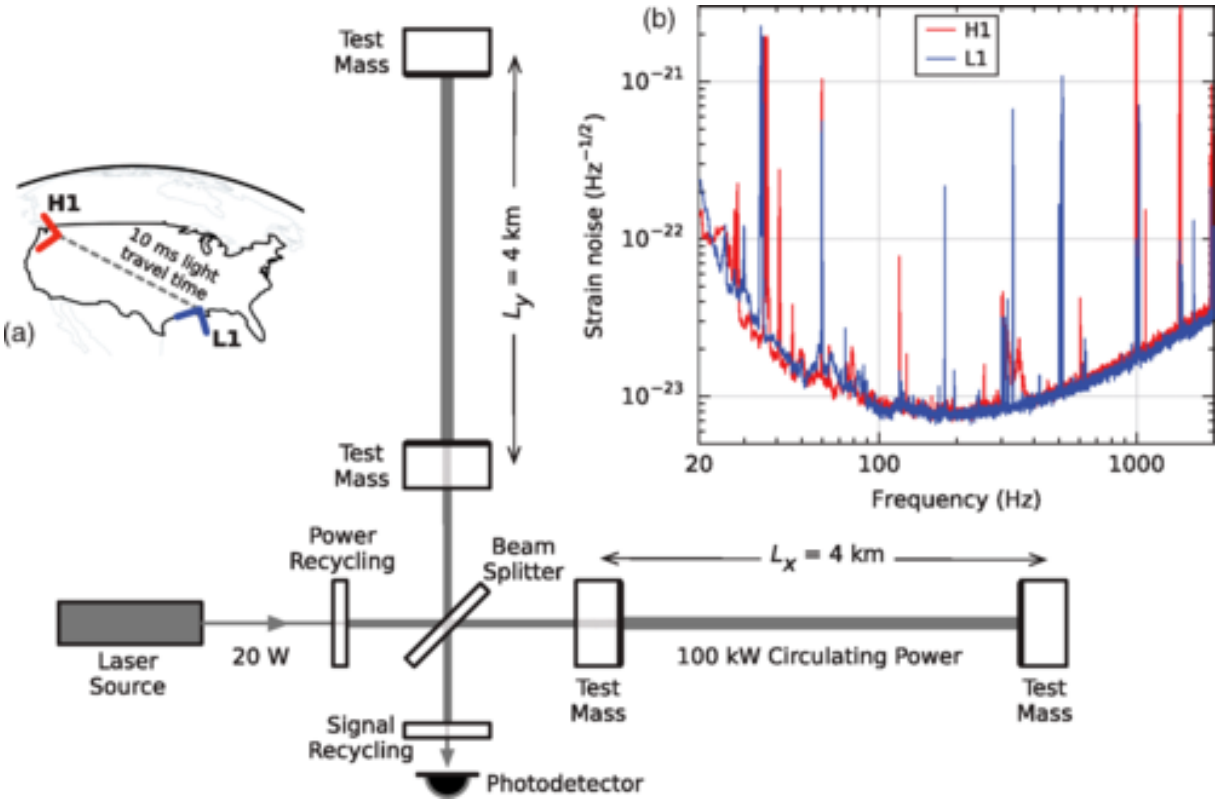


Figure 1.1: Simplified diagram of an Advanced LIGO detector, not to scale (Abbott et al., 2016).

range of detection techniques, going from giant radio antennae on earth to x-ray observatories mounted on satellites, gravitational wave observatories are mostly limited to one technique: laser interferometry.

Figure 1.1 shows the setup of the Laser Interferometer Gravitational-Wave Observatory (LIGO), consisting of two identical detectors on opposite sides of the United States. A laser beam is sent from the source through a beam splitter, producing two beams that travel through the $4\ \mathrm{km}$ long perpendicular arms of the detector to eventually come together again in the photodetector. In an ideal scenario, without gravitational waves, these beams are finetuned to cancel each other out through destructive interference. Should the length of the arms change, the interference will no longer be exactly destructive and the laser signal measured by the photodetector can be translated into the strain, which is the ratio by which lengths are shortened or elongated in the two arms due to the passing of the wave. As a ratio of lengths, the strain is unitless.

Instead of the maximal amplitude of the strain, a GW detection usually records its instantaneous value. Because gravitational wave events are (luckily) so far away, the strain is extremely small, of the order of 10^{-21}, as can be seen in figure 1.2 for the first observed black hole merger event. Apart from a time vs. strain analysis, a gravitational wave can also be evaluated in terms of frequency. Possibilities include a power spectrum, a time vs. frequency plot and a spectrogram. While those last two both have frequency and time on the axes, the latter measures the energy for each frequency as a function of time, instead of just showing which frequencies are present like the time vs. frequency plot.

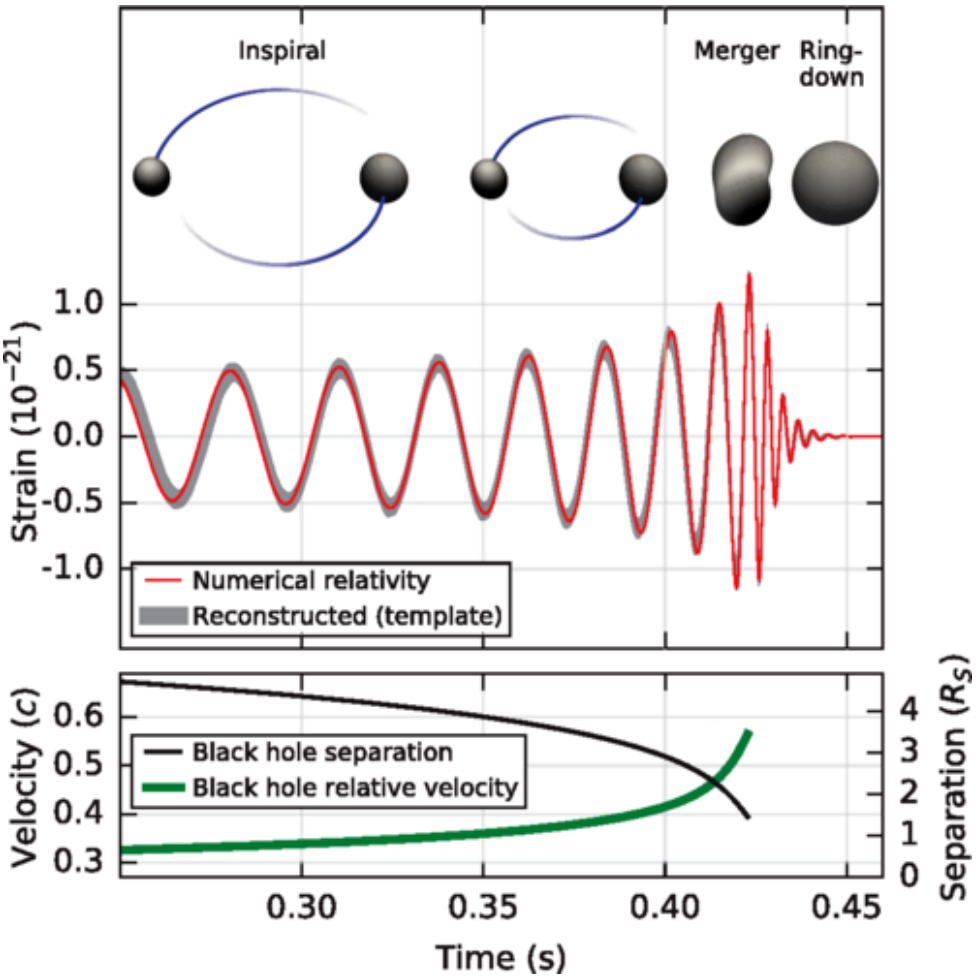


Figure 1.2: Full bandwidth estimated gravitational-wave strain amplitude from GW150914 projected onto H1 (LIGO Hanford), along with illustrations of the binary black hole system (Abbott et al., 2016).

Apart from the two LIGO detectors, two more detectors are simulated in this work. The first one is Virgo, a detector in Italy operated by the European Gravitational Observatory (EGO) collaboration. It has arms of 3 km long and the first joint detection with LIGO was GW170814 (Abbott et al., 2017). The Kamioka Gravitational Wave Detector (KAGRA) in Japan also has 3 km arms. A full joint observing run with LIGO-Virgo-KAGRA is yet to start, but KAGRA was active at the end of the third LIGO-Virgo observing run. Other detectors, like GEO600 in Germany, are not included in this thesis and neither are future detectors like LIGO-India, the Einstein Telescope (ET) and the Laser Interferometer Space Antenna (LISA).

1.2.2 Single object GWs

As opposed to all of the GW events that have been observed to date, supernovae originate in a single stellar object instead of a binary system of black holes or neutron stars. The reason they too are capable of producing gravitational radiation is their asymmetry. A perfectly spherically symmetric supernova explosion, although quite luminous in optical and neutrino observations, would result in no gravitational waves. It is only because of the previously described effects

like SASI, prompt convection and g-modes that large portions of the stellar mass accelerate in an asymmetric way, inevitably resulting in gravitational waves. As the relative velocity of two black holes or neutron stars in a merger far exceed any velocity reached in supernova explosions, it can be no suprise that the GW strain for the latter is also much smaller, hence making them virtually undetectable beyond the local universe.

1.3 Roadmap

After this introduction to the necessary concepts related to supernovae and gravitational waves, Chapter 2 will dive deeper into the techniques that will be used to manipulate the data, from the input of the simulations up to the machine learning algorithms. Chapter 3 will discuss the results of this analysis, followed by some thoughts on improvements in Chapter 4. Appendices A and B contain a summary in Dutch and an in-depth commentary of the source code respectively.

Chapter 2

Methods

2.1 Model selection

For this thesis, a selection of 14 waveform families has been made from various sources. This selection attempts to include as much variety in the characteristics of the supernova and GWs as possible, in order to cover a large volume in the parameter space. Provided below is a short summary for each of these 14 waveform families, so as to get familiar with their common traits and the differences between them and highlight the characteristics that will play an important role later on [1]. The properties of these waveform families are all explained in more details in the respective papers where they were first published, as well as in Szczepańczyk et al. (2021), where most of them have been presented in a single overview.

In order to become familiar with what these waveforms look like and to see how similar and yet how different they are, a plot is included for each of the models along with their summaries. The plots all have the same horizontal scale, whereas vertically they have only been made to be symmetrical, as their amplitudes differ too much to use a common scale there. The waveforms will look differently in the four different detectors due to differences in detector capabilities and orientations. This has been taken into account in these plots. The detector noise however has not been added yet. Each of the plotted waveforms was randomly selected from the waveforms simulated for the purpose of this thesis.

2.1.1 Parameters

One of the main characteristics dividing the models is their mass, where the given mass always concerns the zero-age main sequence mass. The fact that some masses are recurrent is not mere coincidence, but rather originates in the fact that multiple studies used the same proto-neutron star progenitor from yet another study preceding the both of them. For example, progenitors

[1] Since no single model is more important than any of the others, they are listed here in alphabetical order by the authors of the papers as part of which they were published.

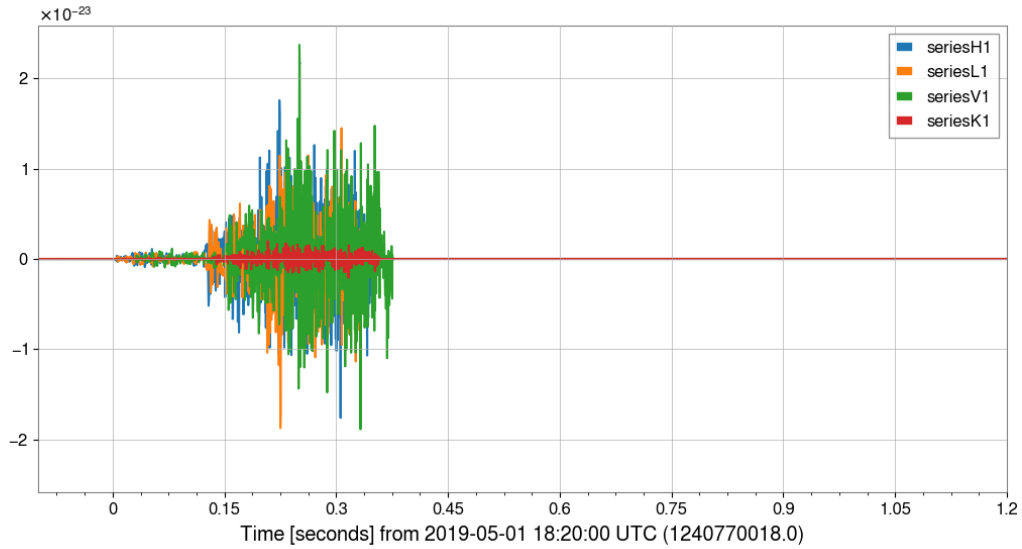


Figure 2.1: An input waveform from the Andresen S11 model.

consisting of $11.2\ \mathrm{M}_\odot$ and $15\ \mathrm{M}_\odot$ are popular, as well as those with a mass of $20\ \mathrm{M}_\odot$ and $27\ \mathrm{M}_\odot$. These specific progenitors are based on the works of Woosley and Weaver (1995), Woosley et al. (2002) and Woosley and Heger (2007).

Of course, as a supernova explosion and its progenitor star can be observed optically as well, the mass might not be the most exciting characteristic to predict using GW observations. That is why, for all models involved, the list of predicted traits not only involves mass, but also the explosion mechanism, the presence of SASI and prompt convection, as well as the rotational velocity.

2.1.2 S11 (Andresen et al., 2017)

The S11 waveform family is the one with the lightest progenitor that is discussed in Andresen et al. (2017). Contrary to its $20\ \mathrm{M}_\odot$ and $27\ \mathrm{M}_\odot$ counterparts from the same study, this $11.2\ \mathrm{M}_\odot$ model does not show strong SASI activity or prompt convection. Non-resonant g-modes are present, whereas resonant g-mode oscillation is suppressed compared to previous (2D) models. The supernovae simulated by this waveform family are neutrino-driven. Over the course of $350\ \mathrm{ms}$ after core bounce, they produce GWs with an energy of roughly $1.1 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$ and a frequency peak around $642\ \mathrm{Hz}$.

2.1.3 M15FR (Andresen et al., 2019)

Andresen et al. (2019) simulated three $15\ \mathrm{M}_\odot$ models, of which the fast rotating one (M15FR) is used here. With an artificially enhanced rotational velocity of $0.5\ \mathrm{rad\ s^{-1}}$ comes also powerful SASI activity, resulting in a higher GW energy than the previous model at $2.7 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$. The neutrino-driven explosion reaches a frequency peak around $689\ \mathrm{Hz}$ during the $460\ \mathrm{ms}$ that were simulated after core bounce. One of the effects of the fast rotation is that unlike in the slowly rotating and non-rotating models, resonant g-modes are present in this model.

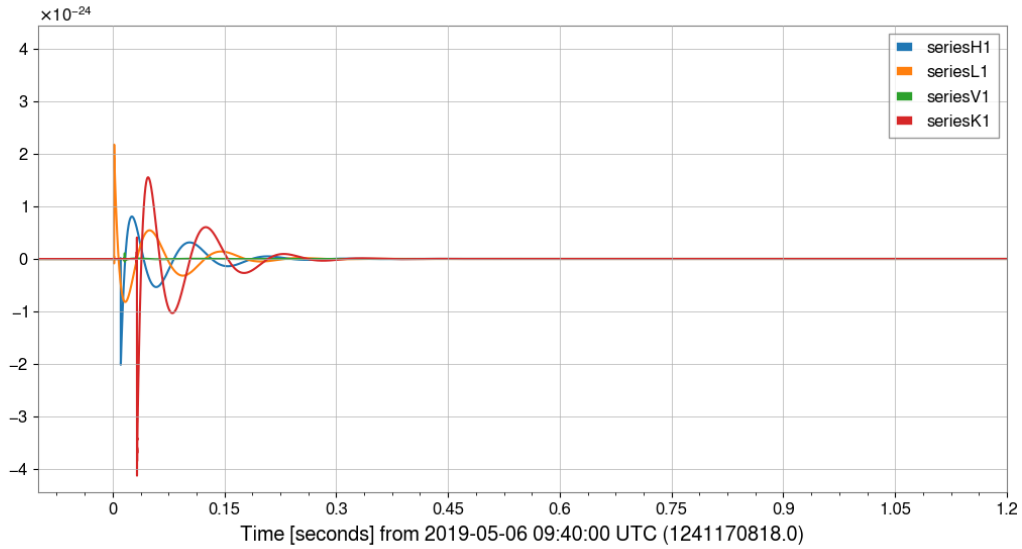


Figure 2.2: An input waveform from the Andresen M15FR model.

2.1.4 H (Bugli et al., 2021)

To explain some of the more energetic supernova explosions, a neutrino-driven scenario does not suffice. Instead, Bugli et al. (2021) combined a fast rotating $35\ \mathrm{M}_\odot$ star with high magnetic fields in order to achieve an MHD-driven explosion.

2.1.5 TM1 (Kuroda et al., 2016)

While there already is a $15\ \mathrm{M}_\odot$ model, this waveform family differs from the one in section 2.1.3 in two aspects. The first difference is that it is non-rotating. The second, more important difference is the equation of state. Kuroda et al. (2016) studied the effect of different equations of state on the supernova characteristics and found that the softer the EoS is, the more the SASI develops. This means that the presence and strength of SASI activity holds information about the EoS as well, which is one of the reasons detecting SASI activity would be quite exciting. Besides SASI activity, g-mode oscillation is also present in the neutrino-driven explosion simulated here. The model reaches a total GW energy of $1.7 \cdot 10^{-9}\ \mathrm{M}_\odot c^2$ in the first $350\ \mathrm{ms}$ post bounce and has a peak frequency of $714\ \mathrm{Hz}$.

2.1.6 S11.2 (Kuroda et al., 2017)

Once again the mass of this model has been selected before and once again there are still a few key differences between this waveform family and the one in section 2.1.2. Kuroda et al. (2017) studied SASI activity, specifically the correlation of its effect on neutrino and GW signals, a topic even beyond the scope of this project. A first difference is the equation of state, which is softer here. This immediately explains why SASI activity was present here, while it was not in the other model of the same mass, as the previous paper by the same authors already showed a correlation between EoS softness and SASI growth (section 2.1.5). Another novelty is that

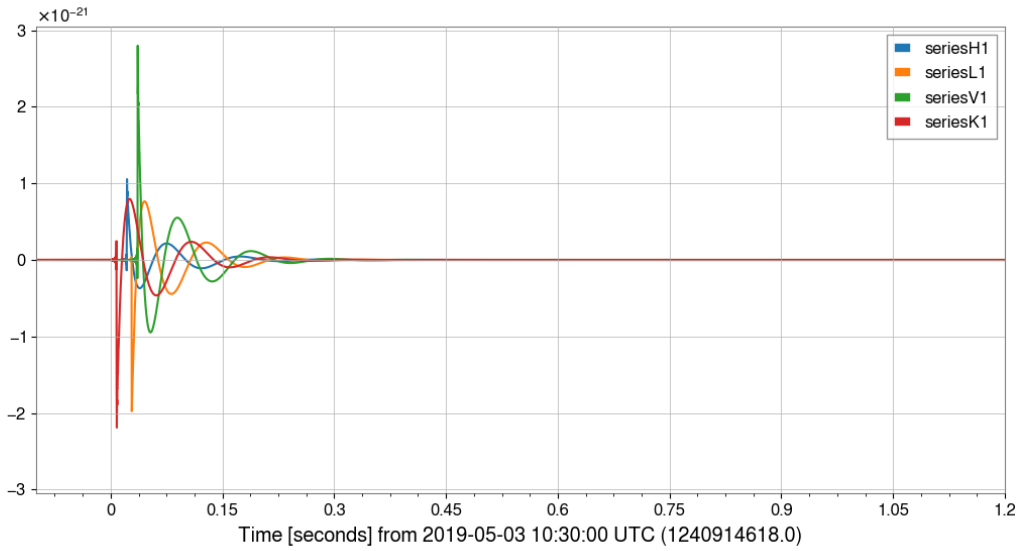


Figure 2.3: An input waveform from the Bugli H model.

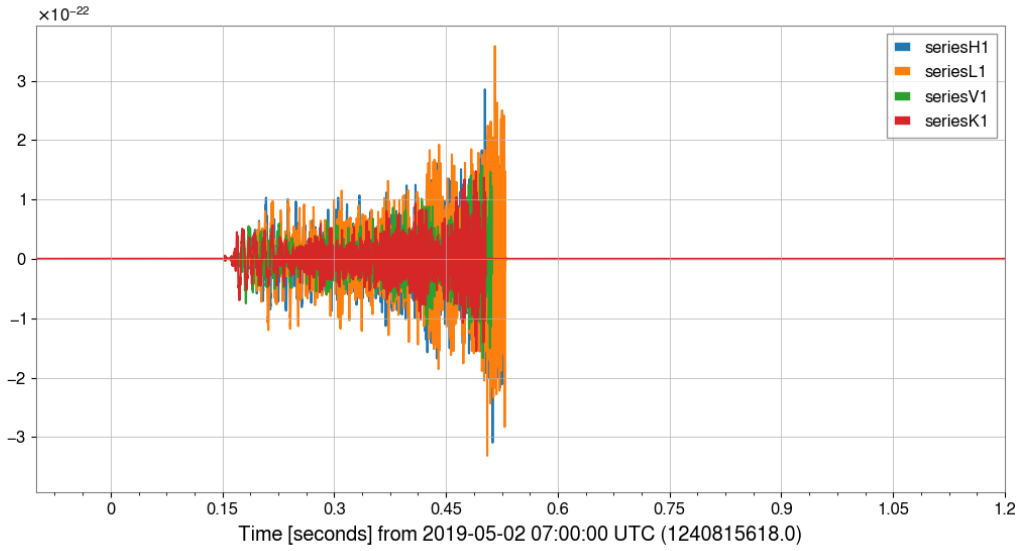


Figure 2.4: An input waveform from the Kuroda TM1 model.

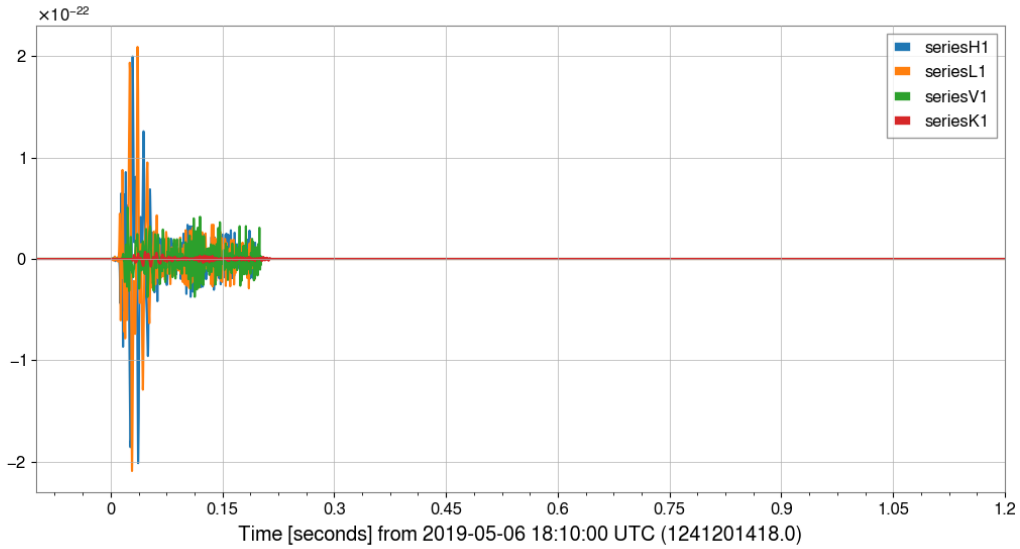


Figure 2.5: An input waveform from the Kuroda S11.2 model.

this non-rotating, neutrino-driven model not only shows g-modes, but also develops prompt convection. Last but not least, in the rather short simulation time of $190\ \mathrm{ms}$ post bounce, the peak frequency reached is $195\ \mathrm{Hz}$ and the GW energy emitted is $1.3 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$.

2.1.7 C15-3D (Mezzacappa et al., 2020)

Mezzacappa et al. (2020) produced yet another $15\ \mathrm{M}_\odot$ model. While, like the model in section 2.1.5, it is a non-rotating, neutrino-driven model exhibiting SASI and g-mode activity, it is still included here. In the $420\ \mathrm{ms}$ simulation, the waveform is significantly more spread in frequency than the other waveforms (Powell et al., 2019) and its peak frequency is also quite high compared to the others: $1064\ \mathrm{Hz}$. Its GW energy lies in the same energy range at $6.4 \cdot 10^{-9}\ \mathrm{M}_\odot c^2$.

2.1.8 Mesa20_pert (O'Connor & Couch, 2018)

The Mesa20_pert simulation by O'Connor and Couch (2018) started from a non-rotating $20\ \mathrm{M}_\odot$ progenitor, but with velocity perturbations added into the silicon and oxygen shell. Apart from that, it is like many others a neutrino-driven explosion with SASI and g-modes present. Its peak frequency is $1033\ \mathrm{Hz}$ and it produces GWs with an energy of $9.5 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$ in the $530\ \mathrm{ms}$ after the bounce.

2.1.9 S27-fheat1.00 (Ott et al., 2013)

Of the four models present in the work of Ott et al. (2013), S27-fheat1.00 is the one with the lowest scaling factor ($f_{\mathrm{heat}} = 1.00$) in the neutrino heating rate. The neutrino heating rate is a term that is of importance in determining if neutrinos can provide energy for the shock wave

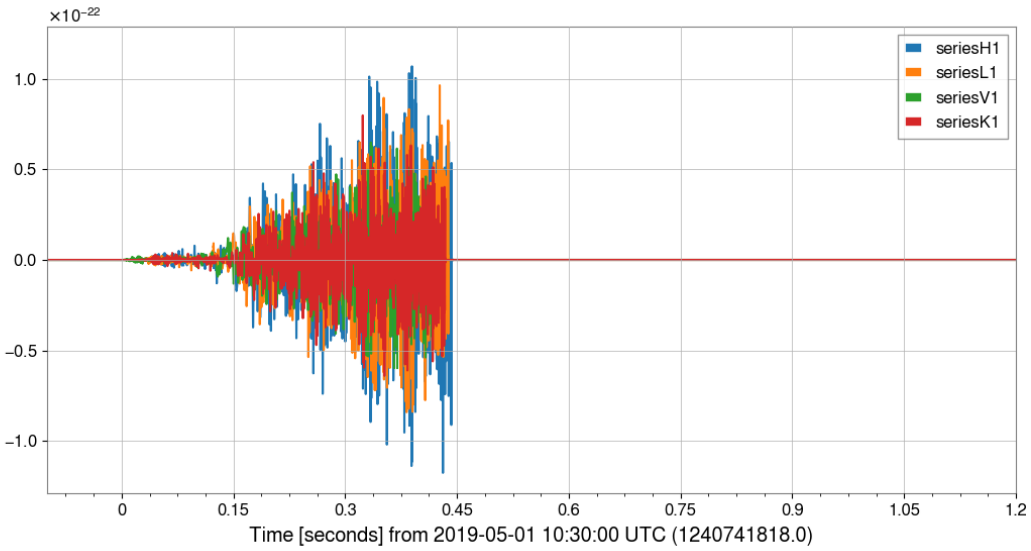


Figure 2.6: An input waveform from the Mezzacappa C15-3D model.

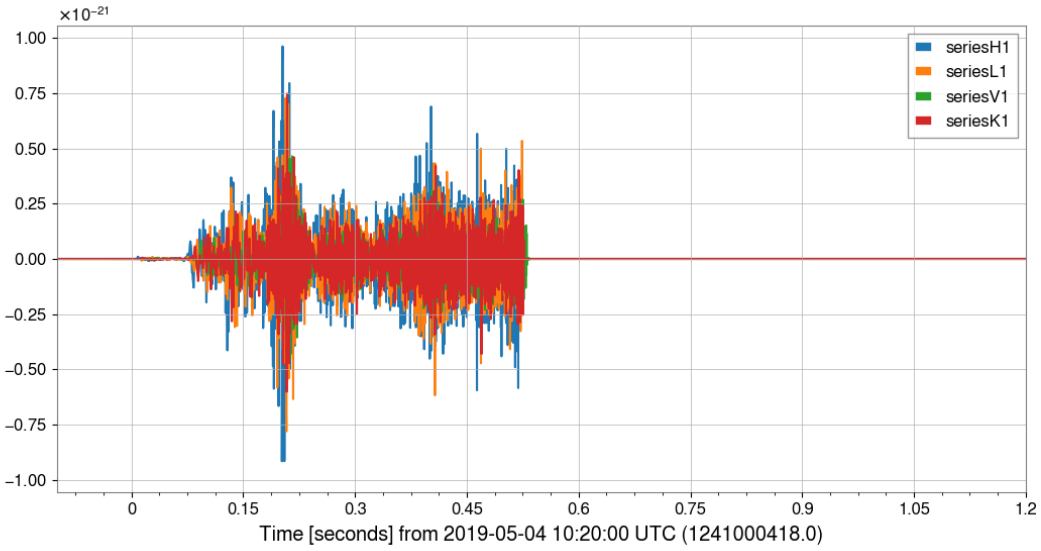


Figure 2.7: An input waveform from the O'Connor Mesa20_pert model.

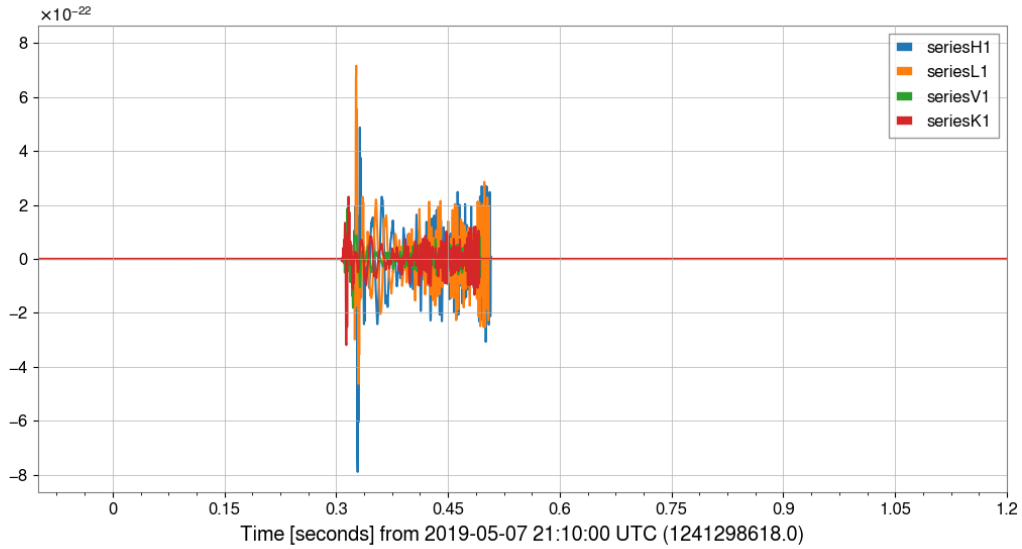


Figure 2.8: An input waveform from the Ott S27-fheat1.00 model.

to continue expanding into a supernova explosion. As the name suggests, the progenitor is a $27\ \mathrm{M}_\odot$ star. As in section 2.1.6, prompt convection shows up here next to SASI activity and g-modes. The non-rotating, neutrino-driven explosion produces $4.0 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$ of energy in $190\ \mathrm{ms}$ post bounce, with a peak frequency of $836\ \mathrm{Hz}$.

2.1.10 S40_FR (Pan et al., 2021)

The S40_FR model has one of the highest masses among the selected models. With its $40\ \mathrm{M}_\odot$ progenitor, it is only exceeded in mass by the model in section 2.1.13. As far as rapidly rotating models go, it is even the heaviest. With a rotational velocity of $1\ \mathrm{rad\ s^{-1}}$, Pan et al. (2021) produced a neutrino-driven explosion with SASI, g-modes and prompt convection present.

2.1.11 S18 (Powell & Müller, 2019)

To study the supernova well into the explosion phase, Powell and Müller (2019) simulated a non-rotating $18\ \mathrm{M}_\odot$ star for $890\ \mathrm{ms}$ after core bounce. The neutrino-driven explosion shows excitation of surface g-modes, but no emission due to SASI activity. The gravitational waves, which have a peak frequency of $872\ \mathrm{Hz}$, are quite strong, as they reach a total energy of $1.6 \cdot 10^{-8}\ \mathrm{M}_\odot c^2$.

2.1.12 M39 (Powell & Müller, 2020)

Whereas their previous model (section 2.1.11) had no rotation, this $39\ \mathrm{M}_\odot$ model is a rapid rotator ($0.542\ \mathrm{rad\ s^{-1}}$). Due to a change in perturbations applied, this time the SASI does develop and so does prompt convection. One of the findings of Powell and Müller (2020) is

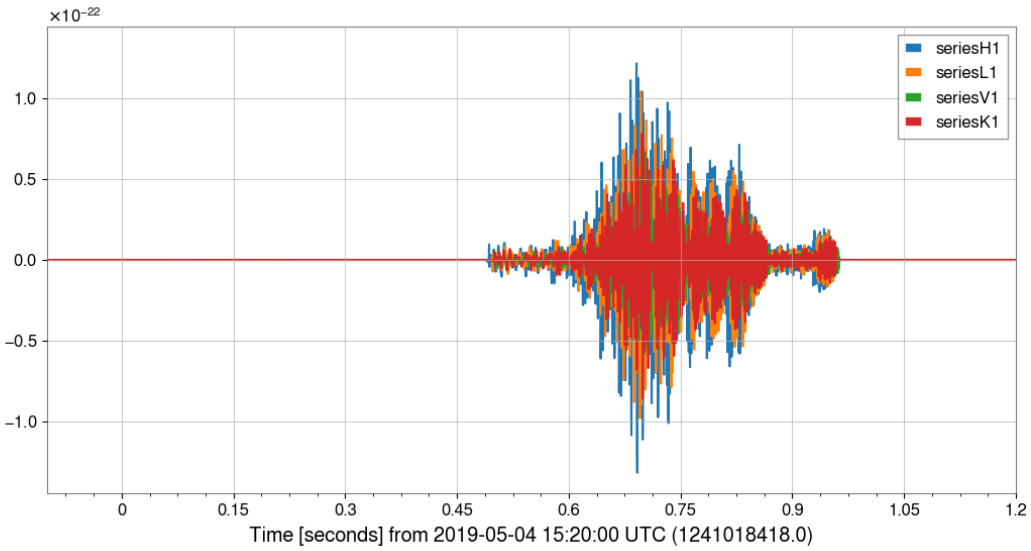


Figure 2.9: An input waveform from the Pan S40_FR model.

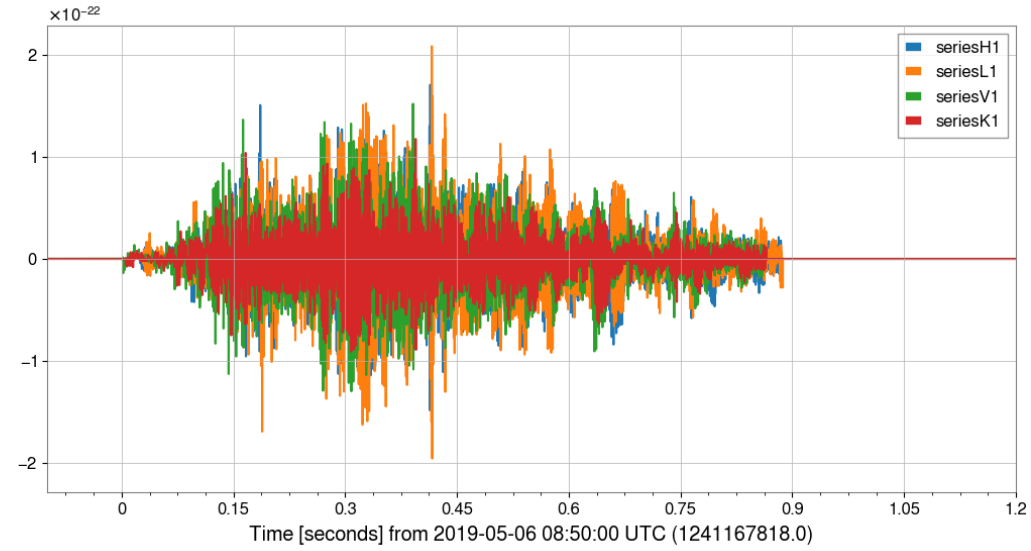


Figure 2.10: An input waveform from the Powell S18 model.

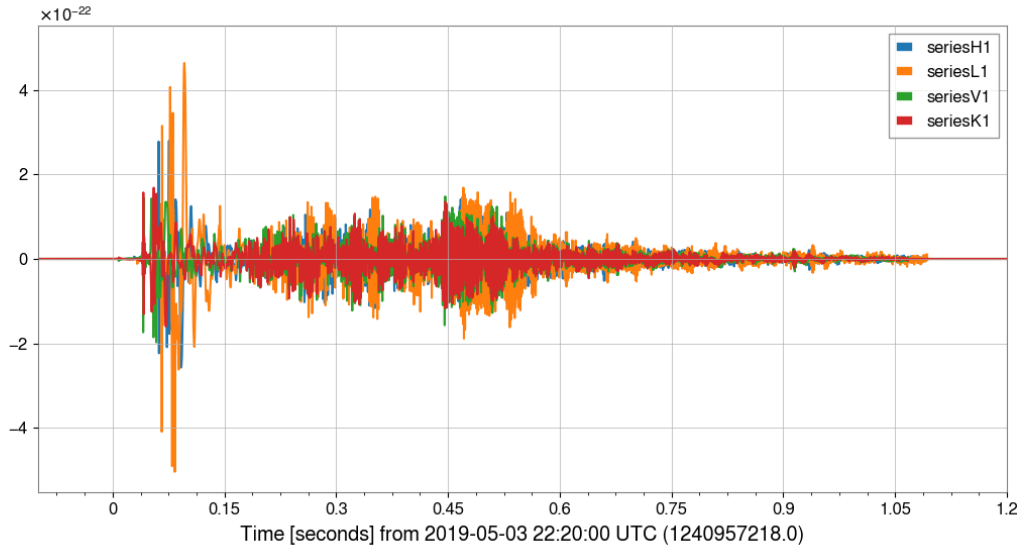


Figure 2.11: An input waveform from the Powell M39 model.

that despite using the neutrino explosion mechanism, the rotation and SASI aid in producing gravitational waves which they claim to be strong enough to be detectable out to $2\ \mathrm{Mpc}$ with the future Einstein Telescope. In $560\ \mathrm{ms}$, the model produces an energy of $7.5 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$ with a peak frequency of $674\ \mathrm{Hz}$.

2.1.13 Z100_SFHx (Powell et al., 2021)

In this quite recent work, Powell et al. (2021) reach for the high end of the progenitor masses. With a stunning $100\ \mathrm{M}_\odot$ progenitor, they investigate if a supernova can occur in such a heavy star before a black hole forms and how far away the gravitational waves from such an event would be detectable. The neutrino-driven explosion develops strong SASI activity.

2.1.14 S9 (Radice et al., 2019)

After the rather heavy model in section 2.1.13, the non-rotating S9 model has the smallest mass ($9\ \mathrm{M}_\odot$) of the waveform families considered. With a few low-mass, neutrino-driven models, Radice et al. (2019) confirm that for a detection to happen at the next galactic supernova event, the detector sensitivities of the next-generation GW detectors will be needed for stars with relatively low masses. With a simulation extending to $1100\ \mathrm{ms}$ post bounce, it reaches a GW energy of $1.6 \cdot 10^{-10}\ \mathrm{M}_\odot c^2$ and a peak frequency of $727\ \mathrm{Hz}$. SASI activity is not visible in the GW signal, but prompt convection is.

2.1.15 R4E1FC_L (Scheidegger et al., 2010)

Last but not least, the R4E1FC_L model is the second MHD-driven model in the selection. With a rotation speed of no less than $9.4\ \mathrm{rad\ s^{-1}}$, Scheidegger et al. (2010) managed to produce a

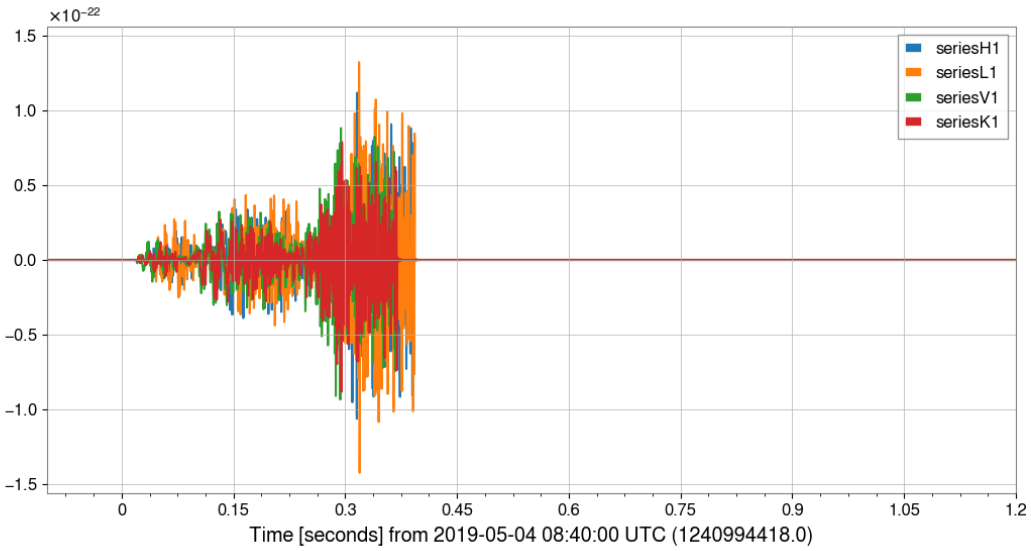


Figure 2.12: An input waveform from the Powell Z100_SFHx model.

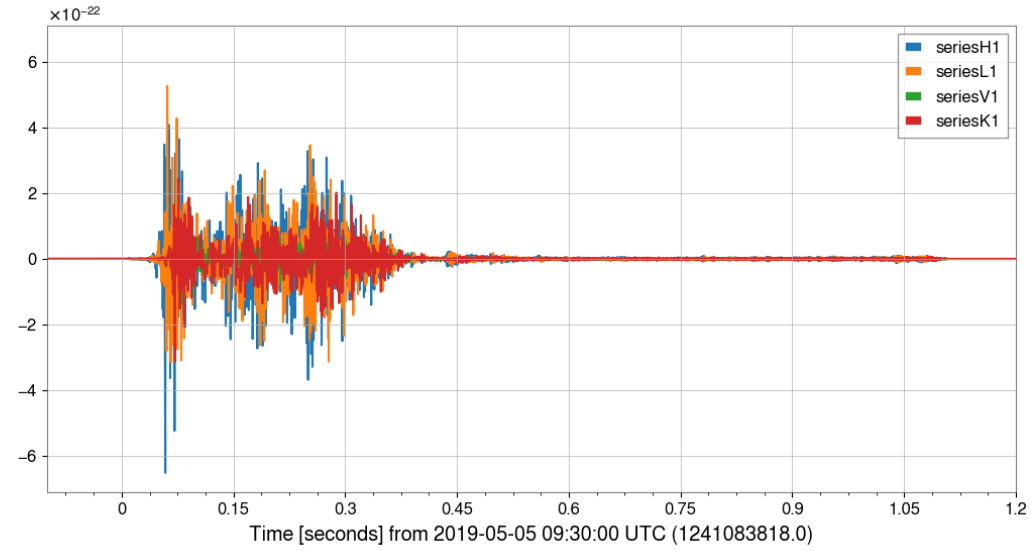


Figure 2.13: An input waveform from the Radice S9 model.

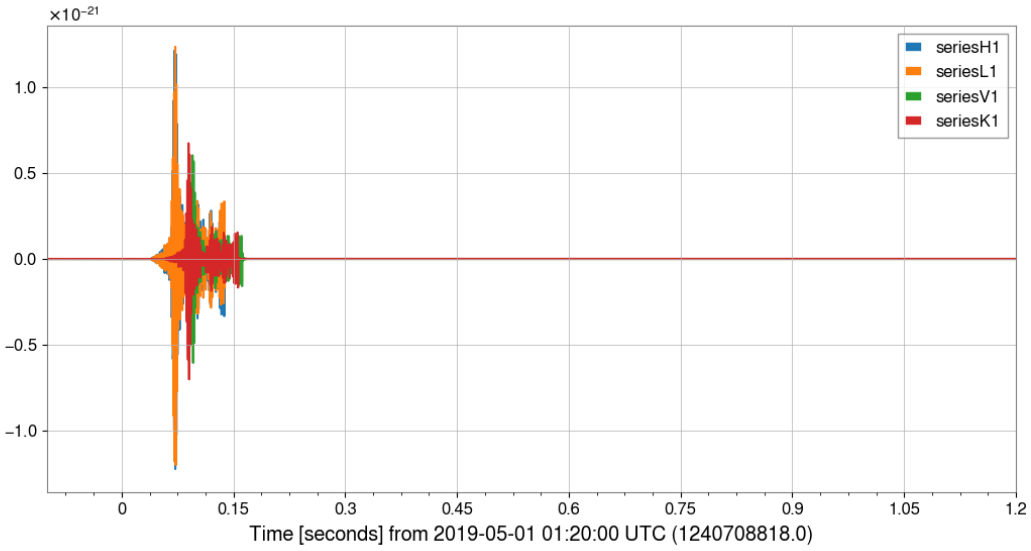


Figure 2.14: An input waveform from the Scheidegger R4E1FC_L model.

model with GW energy equal to $3.9 \cdot 10^{-7}\ \mathrm{M}_{\odot}c^2$. During the rather short simulation (100 ms), they identify prompt convection in the gravitational waves, which have a peak frequency of 683 Hz.

2.2 Model simulation

The simulation of the waveforms is based on datafiles generated by the respective researchers that created the models. These datafiles can be found on an internal LIGO GitLab page. Each of the waveforms can be read in from their respective file and a transformation can then be applied to randomize their incident angle and direction, as well as the distance from which they supposedly originated. This procedure is repeated one thousand times for each of the models. Since all waveforms need to be treated equally and they don't all have the same length, a buffer is added to the front and back of the wave to make each sample 10 s long. The start times of the samples, required to determine the GPS location of the detectors at that time, are separated by 600 s, making the total simulated time for each of the models $6 \cdot 10^5$ s or approximately one week long. This removes any bias that could be introduced by the orientation of the waveforms with respect to the detectors, as in a weeks time the detectors have each rotated along with the earth enough to randomize orientations. Finally, noise must be injected into the signal. For this purpose, stationary, Gaussian noise samples will be used that have an amplitude spectral density resembling the predicted sensitivity curves of the detectors for observing run 5 (O5). For the HLVK[2] network used here, these sensitivity curves are presented in figure 2.15.

Previous studies like Szczepańczyk et al. (2021) have focussed on determining a maximum range to which CCSNe would be detectable with the currect detectors. However, the sensitivities of the detectors change with consecutive upgrades, extending these limits as they go. Instead of

[2]LIGO **H**anford, LIGO **L**ivingston, **V**irgo, **K**AGRA

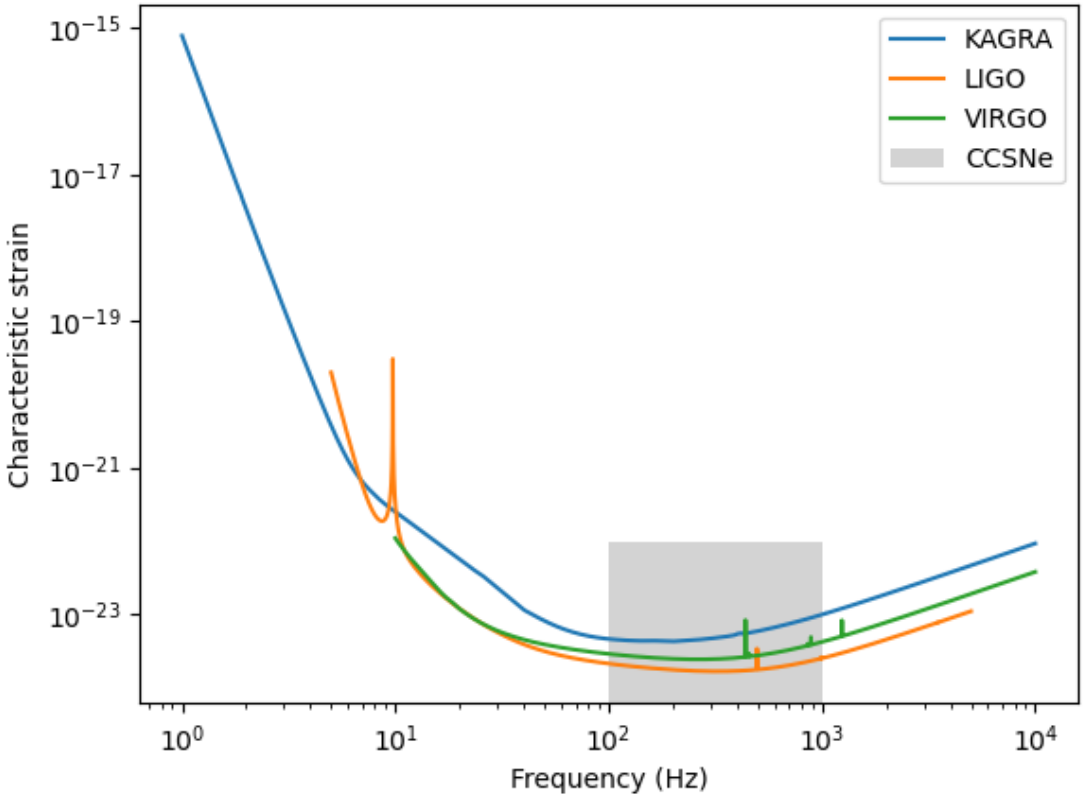


Figure 2.15: O5 detector sensitivities for LIGO, Virgo and KAGRA. The LIGO curve represents both LIGO detectors. The main frequency area where CCSNe are expected to be seen is indicated in grey.

using random distances to see how the machine learning algorithms perform as a function of distance, a transition will be made here to use random signal to noise ratios instead. For this purpose, another transformation must be made. The signal to noise ratio (SNR) for the simulated waveforms depends, among other things, on the distance that was simulated, as well as the noise generated by simulating the passing of the waveform through the detectors. The SNR can be calculated using a specialized function from the supernova toolbox (Szczepańczyk, 2020). New SNRs are then generated ranging from 10 to 100. Below an SNR of 10, waveforms have been proven hard to reconstruct at all (figure 2.16), while at an SNR of 100 they can be almost perfectly reconstructed, so going beyond this range would not be meaningful. The waveforms are then transformed to their new SNRs by multiplying the signal with the ratio of the old and new SNR. More details on the implementation of the simulations can be found in Appendix B.

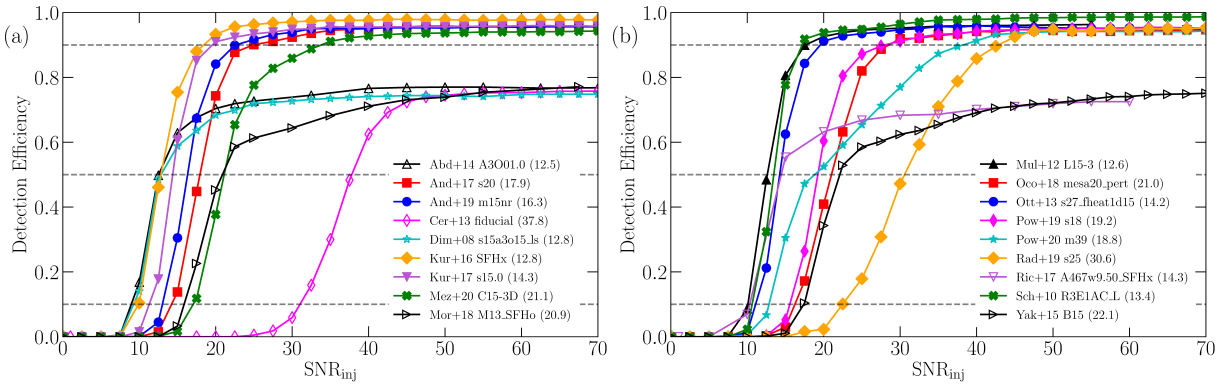


Figure 2.16: Detection efficiency curves as a function of injected SNR for example waveforms from Szczepańczyk et al. (2021) are presented in panels (a) and (b). The numbers in the brackets are SNRs at 50% detection efficiencies. All models used in this thesis reach nearly 100% detection efficiency at the highest SNRs.

2.3 BayesWave

BayesWave is a piece of software designed specifically to deal with gravitational wave observations and the non-stationary, non-Gaussian noise that is inherently part of the detections. The name "BayesWave" is derived from a concatenation of "Bayesian" and "wavelets" (Cornish & Littenberg, 2015). BayesWave will be configured here to use wavelets or wave packets instead of chirplets, which are used in the analysis of a binary coalescence. Chirplets have a longer left tail and are a better fit to events like the one in figure 1.2, while wavelets are better suited for short bursts, such as what is expected from CCSNe. BayesWave uses a Reversible Jump Markov Chain Monte Carlo (RJMCMC) algorithm, varying both the number of wavelets used and the parameters of said wavelets simultaneously. The success of such an algorithm is also influenced by the chosen wavelet basis, which is the Morlet-Gabor wavelet basis for BayesWave. The Morlet-Gabor wavelet basis is an overcomplete basis in which the wavelets are made up of an exponential carrier multiplied by a Gaussian envelope[3].

The Bayesian side of the algorithm is embodied by the use of Bayes' rule. Using this theorem, three components are needed to calculate the properties of a waveform $\mathbf{h}$ given data $\mathbf{s}$ under model M. The first is the prior distribution $p(\mathbf{h}|M)$, which takes into account the assumptions made about $\mathbf{h}$ in the model. The second is the likelihood $p(\mathbf{s}|\mathbf{h}, M)$, which determines how likely it is to see the data if the assumptions about the waveform and the model are correct. The third and last component is the marginal distribution $p(\mathbf{s}|M) = \int p(\mathbf{h}|M)p(\mathbf{s}|\mathbf{h}, M)d\mathbf{h}$, which quantifies how well the data fits into the model, regardless of what waveform is at play. These three components can then be used to calculate the posterior distribution $p(\mathbf{h}|\mathbf{s}, M) = \dfrac{p(\mathbf{h}|M)p(\mathbf{s}|\mathbf{h}, M)}{p(\mathbf{s}|M)}$, which holds all information on $\mathbf{h}$ that can be gathered from $\mathbf{s}$ given M. The purpose of this analysis is to try to retrieve the signal $\mathbf{h}$ from the data $\mathbf{s}$, where the latter is

[3] $\Psi(t; A, f_0, Q, t_0, \phi_0) = A \exp\left(-(t-t_0)^2/\tau^2\right) \cos\left(2\pi f_0(t-t_0) + \phi_0\right)$ with amplitude A, quality factor Q, central time t_0, central frequency f_0, phase offset ϕ_0 and $\tau = Q/(2\pi f_0)$

defined as $\mathbf{s} = \mathbf{R} \cdot \mathbf{h} + \mathbf{n}$, with $\mathbf{n}$ the instrument noise and $\mathbf{R}$ the operator that describes the response of the detector network to a particular gravitational wave signal.

Since BayesWave is computationally more expensive than other GW search algorithms such as coherent WaveBurst, it is mostly used only on a preselected subset of data instead of as the primary search algorithm (Abbott et al., 2021). While in this thesis only BayesWave is used, in a real setup there would indeed be another algorithm preceeding it in the detection pipeline, leaving BayesWave only the task to reconstruct a sample that has already been identified as significant. This preceeding algorithm could be based on BayesWave, as was the case in Dálya et al. (2021), and ensures that glitches are subtracted without compromising the signal. As a result, the remaining noise can be modelled accurately as stationary, Gaussian noise, as will be done in this thesis. Although the noiseless waveforms are available here as they were the result of a simulation, this will not be the case for a real detection, so the data used here consists of the waveforms with noise added after reconstruction using BayesWave. That way, the machine learning algorithms described below will learn to classify the processed waveforms, as they will need to do for a real sample. Raza et al. (2022) has investigated possible optimizations of BayesWaves' parameters specifically for CCSNe. They recommend to use 4 million RJMCMC iterations, a frequency range from 16 Hz to approximately 2 kHz and a segment length of at least 4 s. All of these recommendations will be accepted here.

2.4 Machine learning

Given all these parameters, it is now up to the machine learning (ML) algorithms to try and predict them given the data generated by BayesWave. As stated before, the objective here is not to create a new GW detection algorithm, but rather to determine if some property or characteristic of a supernova can be determined from a supernova induced GW signal, knowing the signal has been identified as a supernova. The detection itself will be done through other methods, quite possibly aided by the simultaneous observation of a neutrino signal and later also by an optical signal [4].

2.4.1 Data selection

The BayesWave reconstructions encompass a large number of data output formats. For each of the 1000 waveforms generated for each of the 14 models, BayesWave will reconstruct the signal as a time vs. strain plot, as well as power spectra, time vs. frequency plots and spectrograms of the data, the median waveform and the residuals. Of these many possibilities, the power

[4] Gravitational waves propagate with the speed of light, as well as both neutrinos and the optical signal itself. The reason why the optical signal is expected to arrive later, is because neutrinos and GWs can escape the collapsing star starting from the beginning of its collapse. The optical signal can only begin its journey towards the detectors once the photons reach the surface of the star at the moment of supernova explosion.

spectrum[5], has been chosen to feed as input to the machine learning algorithms. The power spectrum is the only one of the outputs that does not show time evolution. The time vs. strain output for example shows a clear start and end of the waveform, which may not correspond to the start and end points of an actual detection. Feeding the time vs. strain data to the algorithm could potentially bias the ML algorithm to perform better on data with the same start and end points or, even worse, bias the algorithm to make classifications based on start and end times instead of the actual data. This is not the case for the power spectrum, as it has a power value for each frequency in the frequency range[6], hence there is no start or end point that can impose such a bias.

Once the data has been gathered, part of it needs to be split off to serve as test data, since there is no other way to validate that the algorithms are actually performing well. A train-test split is made, resulting in a training set on which the following steps will be calibrated and a test set. The calibrated algorithms will predict a class for the test set, resulting in a performance score that shows how effective the algorithms are.

2.4.2 Dimensionality reduction

After selecting the input data, the next step is selecting a dimensionality reduction technique. This serves not only the purpose of reducing the amount of data and therefore the time spent training the algorithm, but might also improve the performance by cutting out irrelevant information that could lead to overfitting. However, the data used here consists of very small numbers: the power in the spectrum usually ranges from around 10^{-51} to around 10^{-47}. That's why it is necessary to rescale the data, although rescaling is never a bad idea, even if the data is not unusually large or small. This is done through a simple z-score normalization, setting the mean value to zero and the standard deviation to one. For the dimensionality reduction, two possible techniques will be investigated.

The first technique is a principal component analysis (PCA), implemented in the Python package scikit-learn by Pedregosa et al. (2011). It identifies, one after the other, perpendicular directions in the high-dimensional space that have the largest remaining variance, after which it projects the data onto these principal component directions. An example of this is shown in figure 2.17 for a two-dimensional dataset. This in itself does not reduce the dimensionality, but the data is now stored in such a way that the first few components of each vector representing one data sample hold the bulk of the information. In this transformed dataset, a cut can be made, keeping only the first n components, where n can either be a chosen integer number or it can be determined based on what percentage of the variance of the original data should be explained by the remaining transformed data after the cut.

The second technique is a uniform manifold approximation and projection (UMAP), imple-

[5]The power spectrum can be found in each output folder at `/post/signal/signal_median_frequency_domain_waveform_spectrum_H1.dat`, where H1 should be replaced with L1, V1 or K1 to get the data for each of the four detectors.

[6]The frequencies range from 16 Hz to 2047.5 Hz.

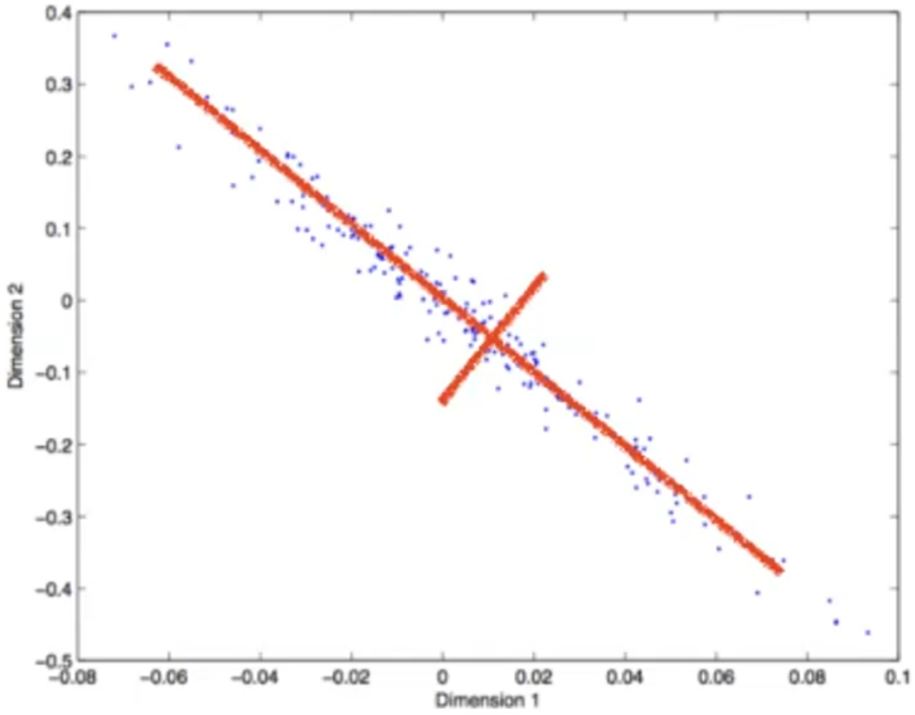


Figure 2.17: Example of a principal component analysis. The line from top left to bottom right represents the first component, the one perpendicular to it represents the second component.

mented in the Python package umap-learn by McInnes et al. (2018). It searches for a representation of the data on a manifold in order to try and reduce its dimensionality. It does this by making use of a weighted k-neighbour graph, for which later a low-dimensional representation is calculated. This method also includes a parameter n to control how many components the final representation should contain.

2.4.3 Classifiers and regressors

With the reduced number of dimensions, the machine learning algorithms can be let loose on the data. A few algorithms will be compared, such as a decision tree (DT), a support vector machine (SVM) and a K-nearest neighbour (KNN) approach for the (binary) classification parameters and a simple linear regression (LR) model as well as a least absolute shrinkage and selection operator (LASSO) model for the numerical parameters.

Decision tree

A decision tree learns to predict the data labels by implementing a number of cuts, each along a single axis in the remaining multidimensional space. It does this in such a way that the impurity of the data is maximally reduced. The impurity, also known as Gini impurity, is defined as $\sum_k p_k(1-p_k)$, with p_k the percentage of samples left on one side of the split for each class k. A normal decision tree will continue until only one class remains in each leaf node, but it is possible to trim the decision tree, for example by setting a minimum value for the impurity decrease, such that smaller decreases will be ignored and certain nodes will not

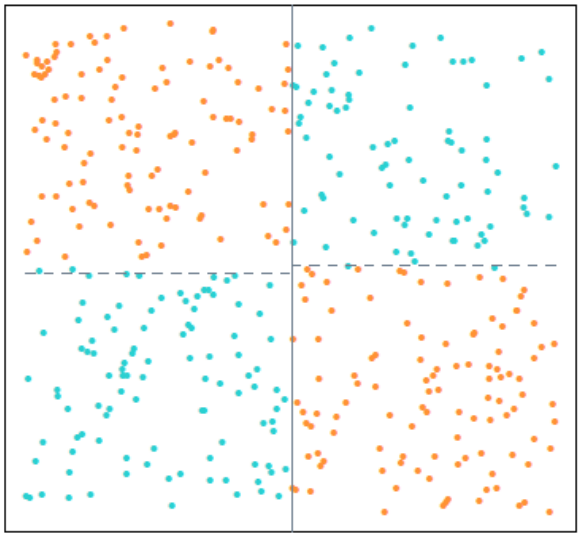


Figure 2.18: Example of a decision tree classifier on an XOR dataset. The dataset consists of blue samples where the x and y axis have the same sign and orange samples where they have an opposite sign. The axes have been removed, but the decision tree splits almost reproduce them.

be split up when they normally would be. This process is known as pruning and may improve performance by preventing overfitting on the training set. Decision trees have the advantage of being easily interpretable. They make consecutive splits, each involving only one dimension. Given the set of parameters for a new sample, it is possible to simply follow these splits down to a node and find the prediction for that sample. This also allows for numerous ways to visualize them, such as in the example in figure 2.18, where a possible DT is visualized by horizontal and vertical lines representing the cuts on an exclusive or (XOR) dataset.

Support vector machine

A support vector machine also makes splits, but does so in all dimensions at once, creating a hypersurface that separates two classes. This hypersurface is positioned in such a way that the support vectors, the margins between it and the closest sample from each class, are as large as possible. For a multiclass classification, a one-vs-the-rest scheme will be used, virtually training 14 SVMs that each distinguish one class from all other classes in the data. A classification is then made based on which of the 14 hypersurfaces a sample is furthest away from, placing it the deepest into the classification area of that specific hypersurface. For example, the black dot in figure 2.19 will be classified as green rather than red, because its distance to the green separator is larger than that to the red one.

K-nearest neighbour

A K-nearest neighbour approach differs from the other two algorithms in that it doesn't actively learn anything about the data. All it does is determine for a given sample what the closest

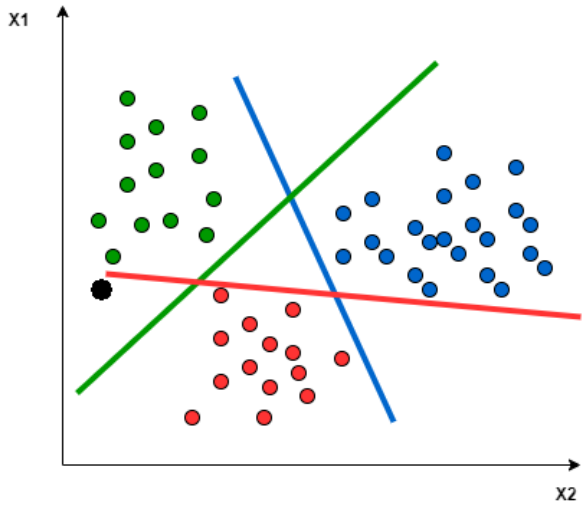


Figure 2.19: Example of a multiclass support vector machine classifier with one unidentified sample (black). The unidentified sample will be classified green with the present separation hypersurfaces.

neighbouring points in the multidimensional training set are, provided a certain distance measure on that dataset. Its prediction is then whichever class is dominant in those k nearest samples, with k a configurable parameter that can be chosen according to the needs of the situation at hand. In binary classifications for example, k should probably be chosen as an odd number in order to prevent a draw. In multiclass cases, k should be chosen high enough to allow one class to take the upper hand instead of ending up with one sample from each of a few different classes.

Linear regression

For predicting the numerical data, the classifiers above are not appropriate. Although there are only a discrete number of values present in the data for both mass and rotational velocity, the quantities themselves are continuous and thus need to be predicted in a continuous way. A first approach to this is a simple linear regression. Predictions consist of the output of $\hat{y} = \sum_i w_i d_i$, where d_i are the data points of the sample and w_i are the weights for each data point in a sample, trained by minimizing the residual sum of squares over the training set.

LASSO

As opposed to the residual sum of squares $\sum_{j=1}^{n}(y_j - \hat{y}_j)^2$ for predictions $\hat{y}_j$ for a training set of n samples with true values y_j, the LASSO method adds an extra term to the error calculation. It minimalizes $\sum_{j=1}^{n}(y_j - \sum_i w_i d_{ij})^2 + \lambda \sum_i w_i$, where the last term serves as a penalty to keep the weights from growing excessively large for any one data point in a sample. This also makes the solutions nonlinear in the training set. The penalty is yet another way to keep the algorithm from overfitting and should in theory result in an improvement over linear regression.

Chapter 3

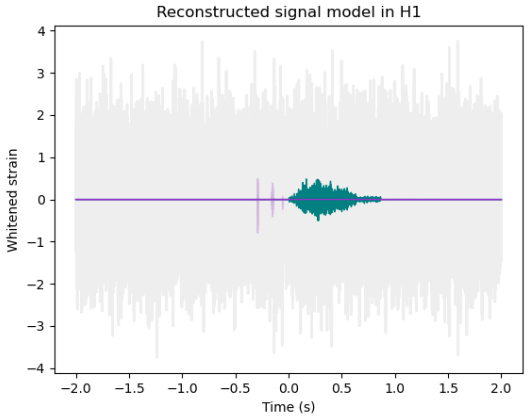
Results

3.1 BayesWave

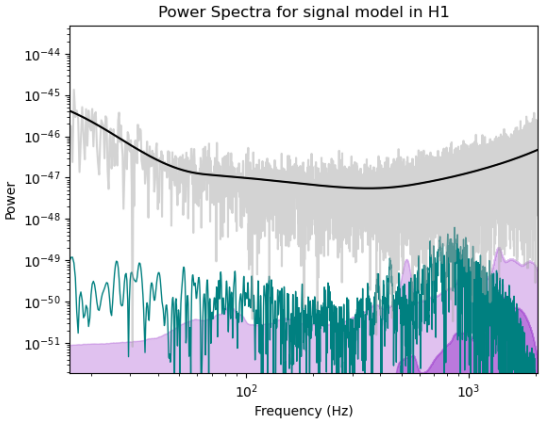
The first type of results that were generated were the reconstructions made by BayesWave. As mentioned before, BayesWave produces a large amount of data, most of which is not used further on. To understand what exactly BayesWave produced, figures 3.1 and 3.2 show examples of both the time vs. strain plots and the power spectra for two waveforms of the Powell S18 model (section 2.1.11). The plots contain three colors: green, purple and grey. The green parts represent the injected signals, while the noise is represented in grey. BayesWave had access to these separate contributions as it had access to the noiseless waveforms. Purple then indicates which parts of the injected waveform BayesWave was able to recover. The difference between low and high SNR cases is clear in the images.

3.2 Dimensionality reduction

While constructing the dimensionality reduction techniques, a few preliminary tests were run on a subset of the data to see if and how well the different models would be separable. The result can be seen in figure 3.3. For both techniques, this seemed promising, as for this particular subset of the data, the models could be separated in only two dimensions. For the full dataset, more than two or even three dimensions were needed to fully separate the models with PCA, which unfortunately cannot be represented in a plot. The same was true for UMAP, but it already showed a decent separation in the first two dimensions, for example in figure 3.4. Comparison of the number of dimensions before and after the reduction revealed a difference by a factor of 45, retaining approximately 730 components as opposed to the 32508 components each sample was represented by before processing.

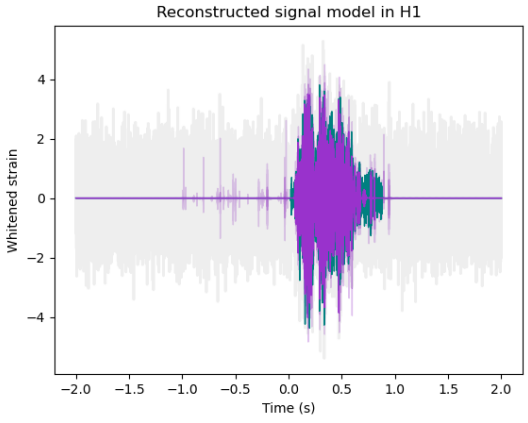


(a) Time vs. strain waveform

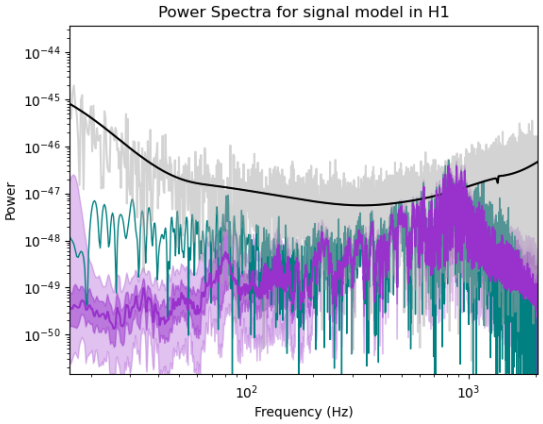


(b) Power spectrum

Figure 3.1: BayesWave reconstruction of a waveform of the Powell S18 model with a low SNR (11.810).



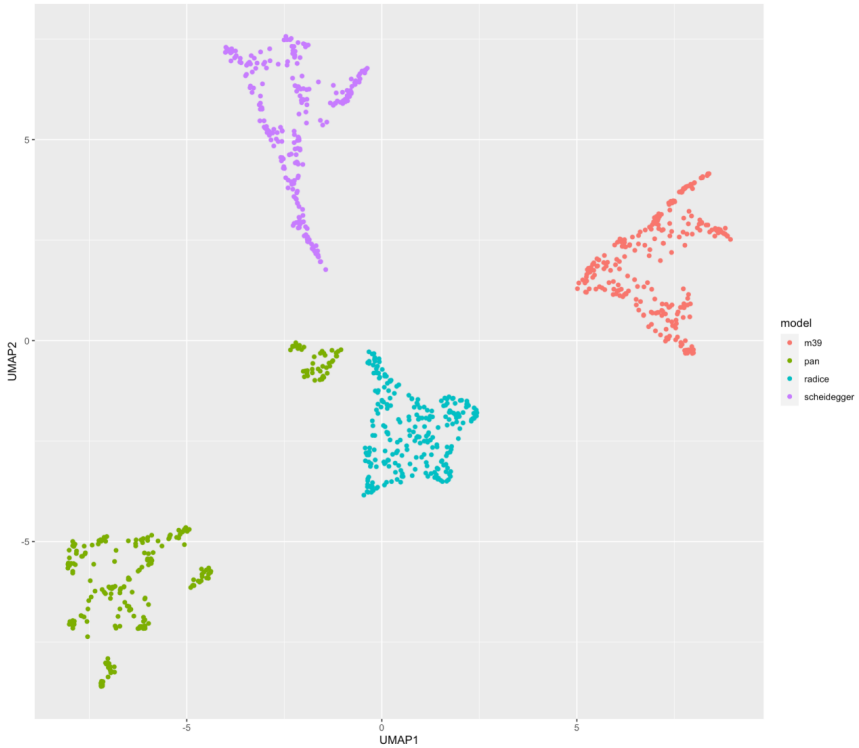
(a) Time vs. strain waveform



(b) Power spectrum

Figure 3.2: BayesWave reconstruction of a waveform of the Powell S18 model with a high SNR (90.692).

(a) PCA



(b) UMAP

Figure 3.3: Preliminary results of dimensionality reduction on a subset of the data involving four models: Powell M39 in red, Pan S40_FR in green, Radice S9 in blue and Scheidegger R4E1FC_L in purple. For both techniques, the axes correspond to the first two components.

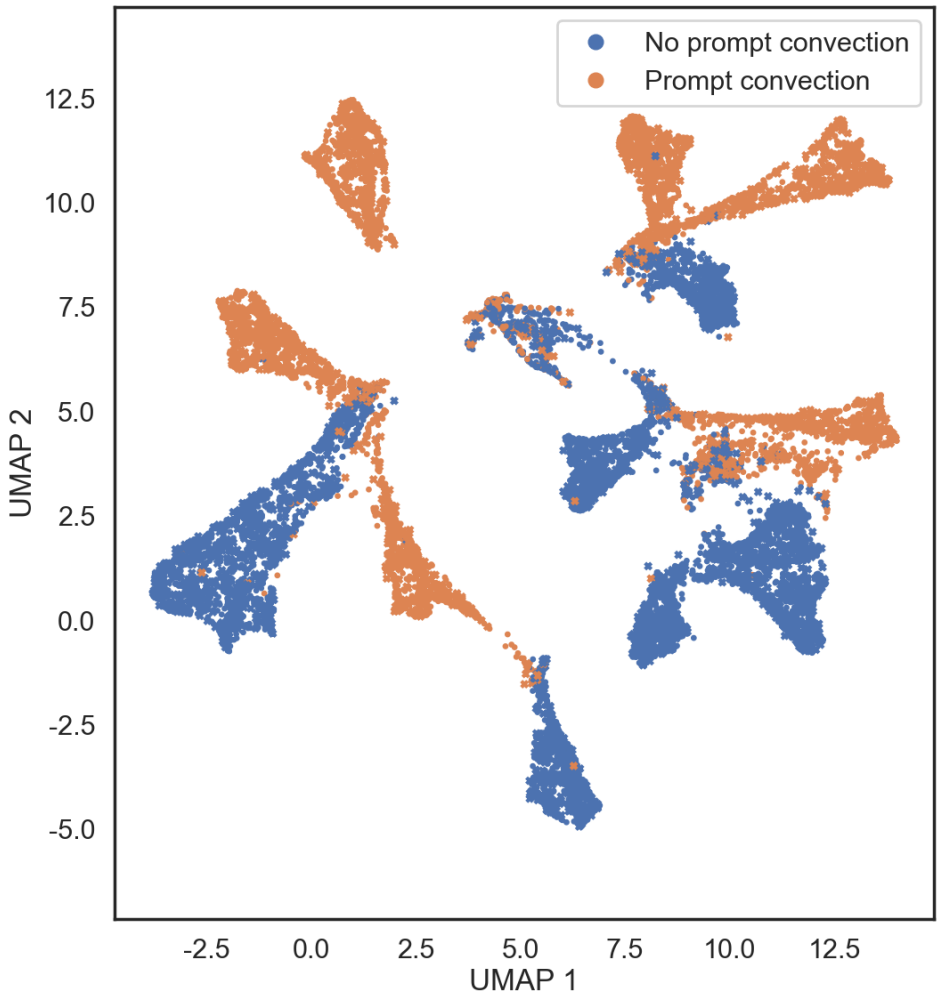
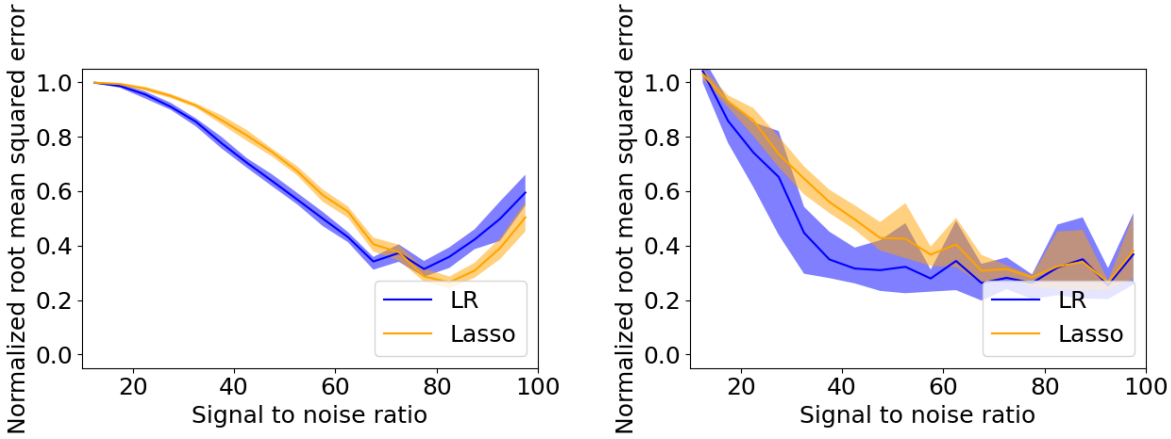


Figure 3.4: UMAP representation of the entire dataset, colored by the presence of prompt convection

3.3 Regression

In order to compare the different algorithms used, there was a need for an accuracy measure. For the numerical labels, this function was chosen to be the normalized root mean squared error (NRMSE). Its definition is $NRMSE = \frac{1}{\sigma_O}\sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_{i,O} - y_{i,P})^2}$, with the differences involving the original (O) and the predicted (P) set of parameters and n samples in the test set with a standard deviation σ_O. This is related to the penalty function used by linear regression and LASSO during training. The fact that it is normalized also allows for a comparison across algorithms, and if desired also across predicted parameters, to see where the predictions are the most accurate. The value of this error metric has been plotted against the signal to noise ratio, which allowed investigating what its impact was on the accuracy of the prediction. For each parameter and for each dimensionality reduction technique, this plot shows the average performance of the predictor. The shaded areas correspond to the standard deviation on this average, calculated on 10 distinct executions of the algorithms. The samples were binned in SNR bins of width 5 before calculating the error.



(a) PCA

(b) UMAP

Figure 3.5: Normalized root mean squared error on the predicted progenitor mass as a function of SNR for both dimensionality reduction techniques.

3.3.1 Mass

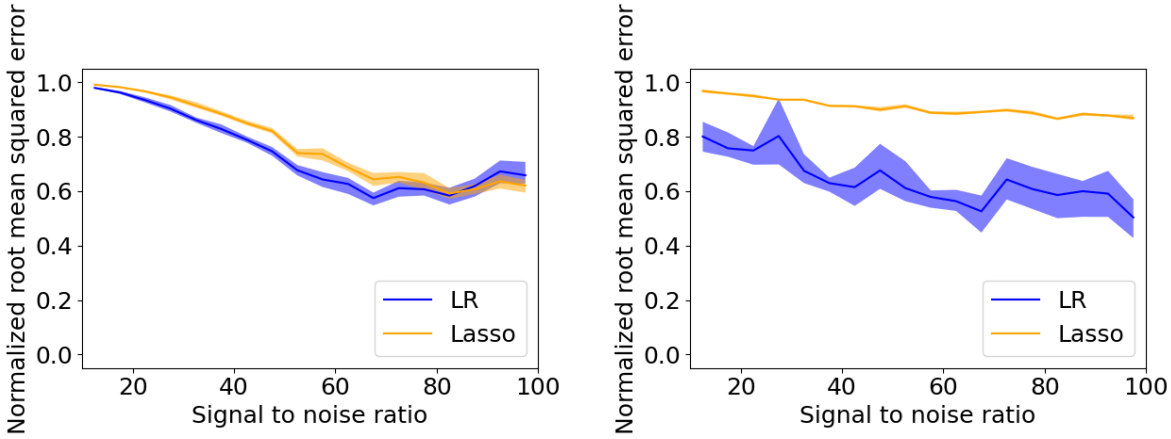
For the case of mass predictions, figure 3.5 suggests that the linear regression was slightly more accurate. The expected drop of error with increasing SNR is present, at least at first, but the error goes up again for some reason at higher SNR. This issue seems to be less pronounced when using UMAP, but the curve still flattens out instead of continuing its downwards trend.

3.3.2 Rotational velocity

While for mass predictions UMAP outperformed PCA, this was no longer the case for the predicted rotation velocities (figure 3.6). Both machine learning algorithms in combination with both dimensionality reduction techniques performed worse than their counterparts for mass predictions.
A trend that can be seen across the numerical algorithms is that the PCA-based predictors are showing a more stable behaviour, while the UMAP-based ones are showing more variance on their results.

3.4 Classification

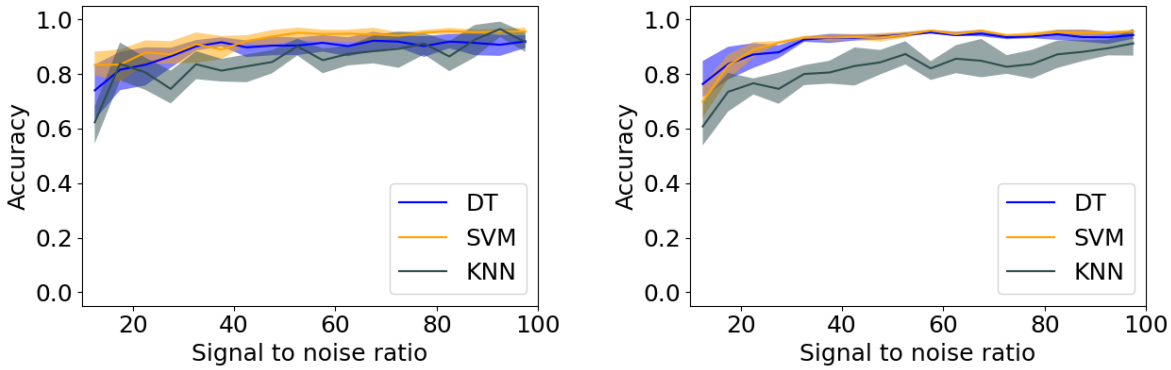
As the characteristics that are predicted here could not be represented numerically, a different performance measure had to be chosen. Because there weren't an equal amount of samples for each class for the different characteristics, the balanced accuracy was used. It is defined as the recall for each class separately, averaged out over the different classes. Recall (R) for a class C is defined as $R = \frac{\sum_{i=0}^{n}(y_{i,O} = C \& y_{i,P} = C)}{\sum_{i=0}^{n}(y_{i,O} = C)}$, the number of correctly predicted members of class C divided by the total number of samples of class C. This would make the balanced accuracy $BA = \sum_{j=1}^{m} \frac{\sum_{i=0}^{n}(y_{i,O} = C_j \& y_{i,P} = C_j)}{\sum_{i=0}^{n}(y_{i,O} = C_j)}$, where $m = 2$ for all cases considered here.



(a) PCA

(b) UMAP

Figure 3.6: Normalized root mean squared error on the predicted progenitor rotational velocity as a function of SNR for both dimensionality reduction techniques.



(a) PCA

(b) UMAP

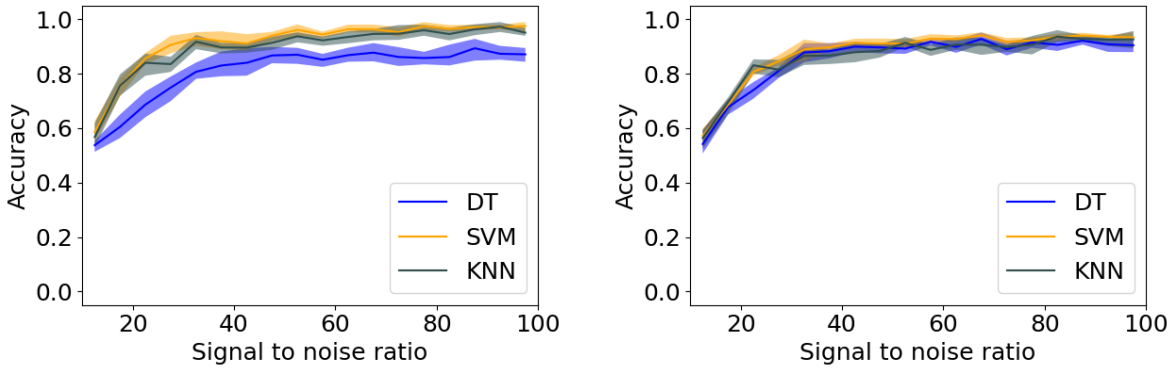
Figure 3.7: Balanced accuracy of the predicted explosion mechanism as a function of SNR for both dimensionality reduction techniques.

3.4.1 Explosion mechanism

With both dimensionality reduction techniques, the KNN classifier achieved a poorer performance than the decision tree or the SVM, as evidenced by figure 3.7. Keeping the mechanism behind KNN in mind, this suggested that some models with different explosion mechanisms partly occupied the same region of space in the reduced but still high-dimensional dataset and their proximity confused the KNN algorithm. This will be further investigated in section 3.5, where the offending models will be sought out.

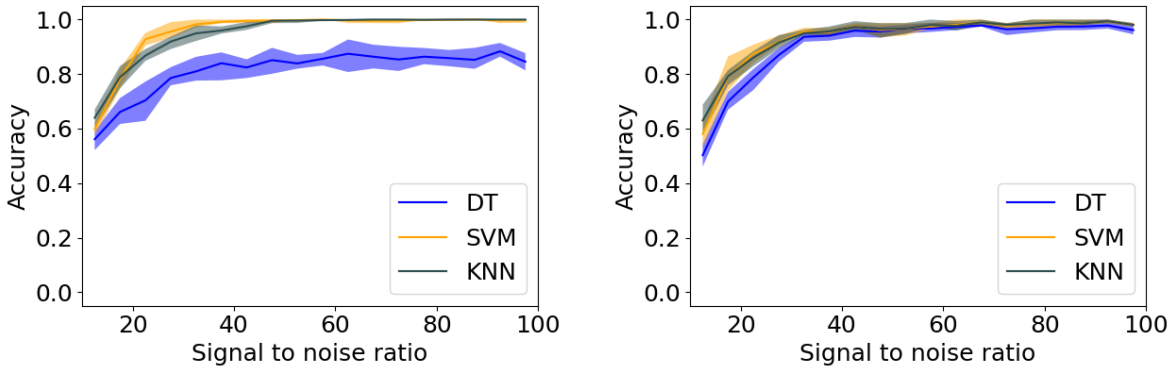
3.4.2 SASI

The three ML algorithms performed quite well, apart from a slight underperformance of the decision tree combined with PCA when compared to the other algorithms. The standard deviations in figure 3.8 are also smaller than they were for the explosion mechanism.



(a) PCA (b) UMAP

Figure 3.8: Balanced accuracy of the predicted presence of SASI as a function of SNR for both dimensionality reduction techniques.



(a) PCA (b) UMAP

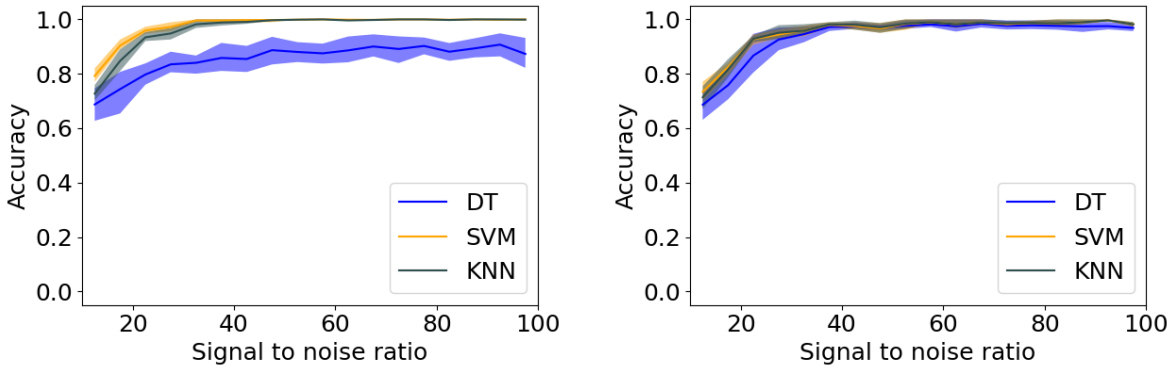
Figure 3.9: Balanced accuracy of the predicted presence of prompt convection as a function of SNR for both dimensionality reduction techniques.

3.4.3 Prompt convection

As was the case for SASI, the PCA-DT combination had more trouble detecting the presence of prompt convection than any other combination (figure 3.9). The SVM and KNN had no problem separating the presence from non-presence for sufficiently high SNR and neither did the UMAP-DT combination, so the issue must have had another source. It might be that the cuts required to separate the classes did not align with the axes of the high-dimensional space, which is a known flaw of the decision tree algorithm.

3.4.4 Rotation

Since the predictions of rotational velocity (section 3.3.2) weren't all that great, a different approach has been taken here. Instead of predicting the numerical value of the rotational



(a) PCA

(b) UMAP

Figure 3.10: Balanced accuracy of the predicted presence of rotation as a function of SNR for both dimensionality reduction techniques.

velocity, the predicted characteristic is the presence or non-presence of rotation altogether. This resulted in figure 3.10, which shows much better results than the numerical analysis.

3.5 Two models confused

Looking at the original waveforms themselves (section 2.1) and the confusion matrix in figure 3.11, it became apparent that the main source of confusion between different models originated in the confusion between the Andresen M15FR (section 2.1.3) and the Bugli H (section 2.1.4) models. Both waveforms are low in the number of distinct parts in the original waveforms, which can lead to them having similar parametrizations in the reduced dimension space. To see what the impact of this confusion is, the machine learning algorithms have been executed a second time, but without including the Andresen M15FR waveform.

Excluding one waveform did indeed improve the results. For example, the increased accuracy for the prediction of the explosion mechanism and the rotational velocity can be seen in figures 3.12 and 3.13 respectively.

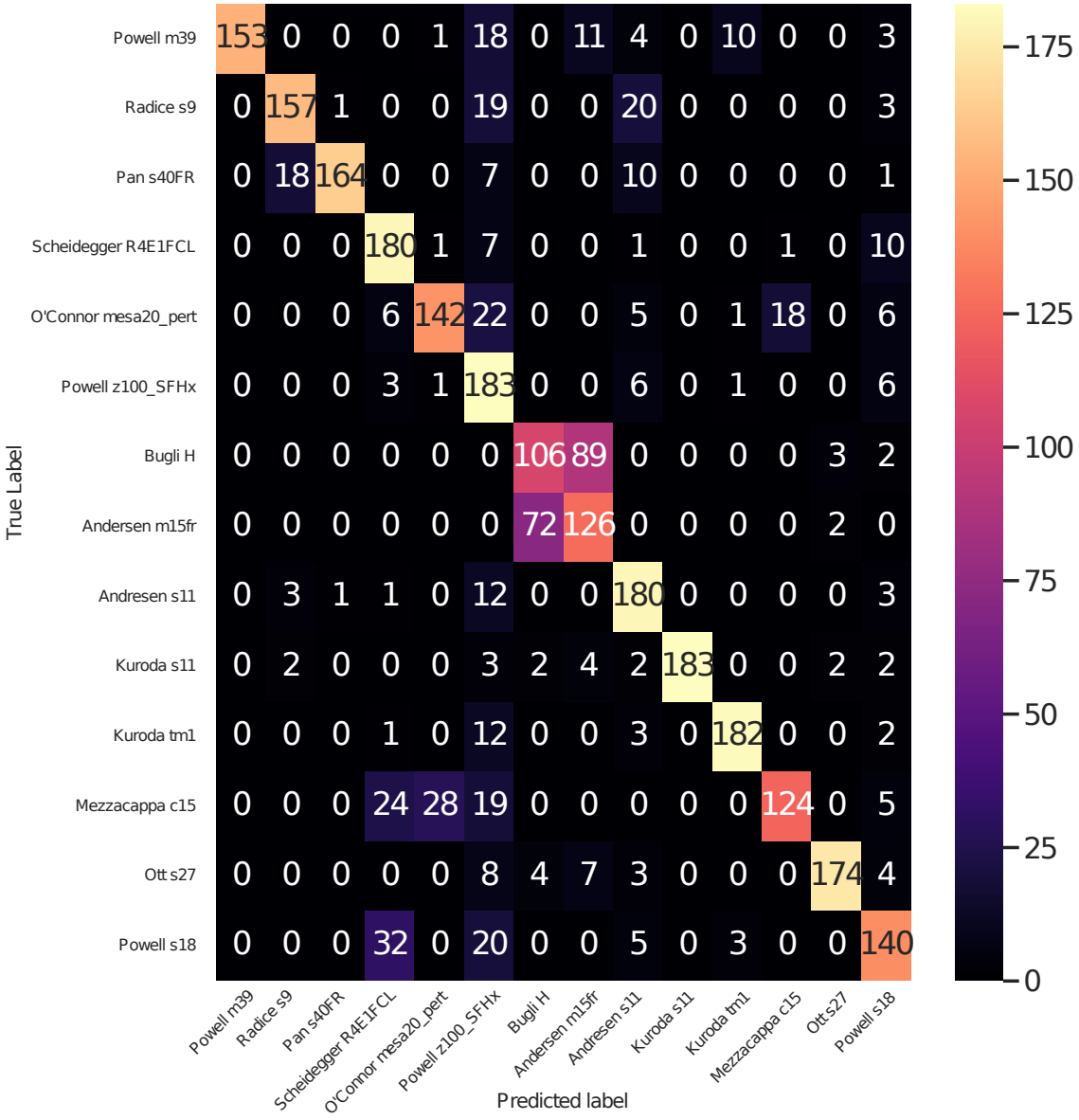
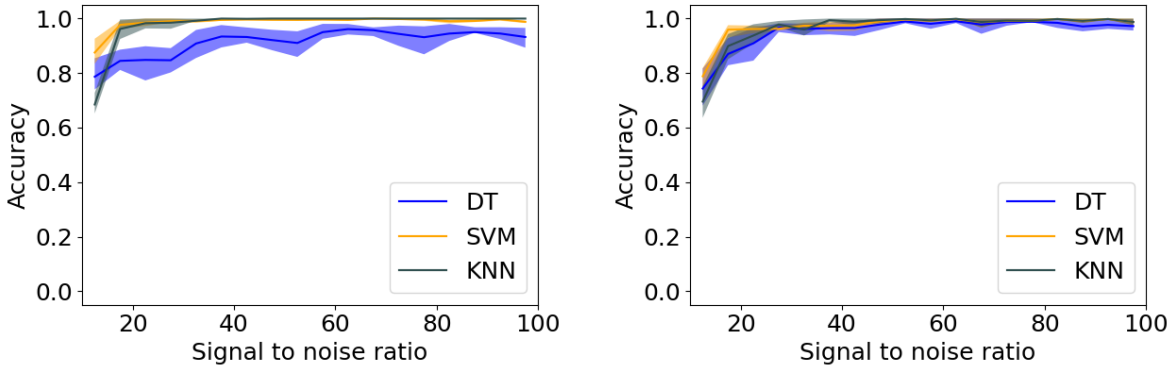
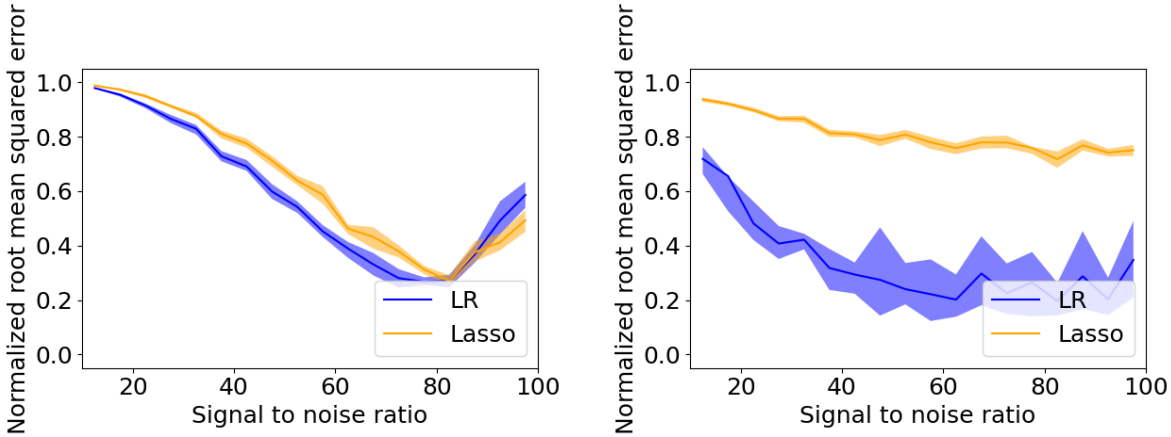


Figure 3.11: Confusion matrix of the identification of separate supernova models. A perfect prediction would render this matrix diagonal.



(a) PCA (b) UMAP

Figure 3.12: Balanced accuracy of the predicted explosion mechanism as a function of SNR for both dimensionality reduction techniques after excluding the Andresen M15FR model.



(a) PCA (b) UMAP

Figure 3.13: Normalized root mean squared error on the predicted progenitor rotational velocity as a function of SNR for both dimensionality reduction techniques after excluding the Andresen M15FR model.

Chapter 4

Future work

This chapter gathers some ideas on improvements and extensions that can be made to the simulations and the further processing thereof, all the way to the machine learning part.

4.1 Physics

4.1.1 Real data

The first and most obvious addition to this work would be its application on a real data sample. Since that doesn't exist yet, this will just have to be put on the sidelines until the Milky Way happens to produce its next core-collapse supernova. Until then, preparations will of course continue, certainly by expanding the detector capabilities and perhaps also by an expansion of the techniques presented in this thesis.

4.1.2 New detectors

A second addition which will have to wait until new data becomes available is the inclusion of more detectors. Four detectors were included here, but a few years from now, there may be a need to add the ones that are still in the planning phase now. All that's needed for this essentially boils down to the sensitivity curve. In fact, the sensitivity curves that were used here (figure 2.15) could even be considered as future detectors, as they are the projected O5 sensitivities for the detectors that are now upgrading to prepare only for observing run 4 (O4). The sensitivity curves allowed the simulation to assess which signals are detectable by a certain detector, resulting in the different amplitudes for each detector as seen in figures 2.1 through 2.14. There is no practical difference between a new detector and an update to an already included detector, both come down to including the new sensitivity curve in the simulation process.

4.1.3 New models

Apart from detectors, new supernova models are also being created. While some will have similar features as the ones that are already included, there might be others that can definitely extend the parameter space that is covered. For example, there are currently only two models included with MHD-driven explosions, so there's certainly room for more of those. A few more high-mass models also wouldn't hurt, as well as models with some more different rotation velocities. Adding models requires a datafile with a generic waveform, as well as some coding to format them in the same way the other waveforms are formatted (e.g. random incident angles, $10\ \mathrm{s}$ long samples, etc.). Along with new models, it's also possible to look through the unused existing models and see if some of them extend the parameter space as well. A good place to start looking for those is in the same place the currently used models were found, as their authors rarely simulated only one model at a time. While doing this, the results of section 3.5 need to be kept in mind, meaning that adding models carries a risk of having more and more confusion between two specific models with different characteristics but similar waveforms due to a low number of distinct frequencies.

4.2 Simulation and data preparation

4.2.1 Incremental models

While adding new data to the simulations will probably improve performance, not all additions are equally simple. Adding more samples of an already incorporated model for already incorporated detectors should present the least trouble. The data from different samples only comes together after extraction from the BayesWave reconstructions, so it's just a matter of getting it labeled correctly and that's that. The same is true for adding a new model. In both cases, it will however be necessary to retrain both the dimensionality reduction techniques as well as the machine learning algorithms, except if the extra samples would only be used for testing. This is where the trouble begins, as retraining requires the original training data to still be present. While it can be kept in its extracted form and not necessarily in the original BayesWave output format, data files can still easily take up a few gigabytes worth of storage for the thousands of samples that were generated here. Luckily, there exist methods like incremental PCA[1] for an incremental approach to dimensionality reduction. For the classification or regression part, stochastic gradient descent (SGD) based algorithms[2] can handle adding data at later times and so can neural networks (NNs)[3], since they process all data sample by sample or in small batches anyway.

A completely different problem arises when a new detector is to be added. Since detector response is taken into account during the BayesWave reconstruction and the combination of detector responses is used to increase the reconstruction, it is no longer possible to linearly

[1] `IncrementalPCA` in `sklearn.decomposition`
[2] `SGDClassifier` or `SGDRegressor` in `sklearn.linear_model`
[3] `MLPClassifier` or `MLPRegressor` in `sklearn.neural_network`

add new data. It is possible to add a new detector response in the simulated waveforms, as the detectors are still separated there, but the data for different detectors needs to go through BayesWave together for each sample of every model. Were it not BayesWave, then surely the next steps would raise the same issue, as they too take data from all detectors into account at once. Apart from the first simulation step, adding a detector thus involves virtually starting over. The decision on whether or not to add a detector will therefore involve a trade-off between resources invested and accuracy gained, where the latter remains to be investigated. The former however can already be estimated, as acquiring the BayesWave reconstructions for the 1000 samples for each of the 14 models in this thesis took at least a few weeks of computation time on a multicore computer grid[4]. Compared to that, the few hours it took to extract the data from the BayesWave output and to train the ML algorithms were negligible[5].

4.2.2 Dimensionality expansion

As the data were transformed into a lower dimensional representation, it's hard to visualize what exactly the components represent, even more so because the input data are power spectra instead of the time vs. strain data format that is easier to assign physical meaning to. This makes it hard to grasp what kind of separation the ML algorithms learn, even though most of the algorithms themselves are not that hard to interpret. It would therefore be interesting to try to reverse the transformation and see what the principal components or the UMAP dimensions look like when transformed back into the power spectrum domain or even into the time vs. strain domain, which is after all only a Fourier transform away. As the principal components basically serve as a basis in which all samples can be represented, this might reveal possible improvements to be made in the basis functions used by BayesWave for CCSNe specifically.

4.3 Machine learning

4.3.1 Parameter grid search

A number of the machine learning algorithms used here have tunable parameters that can significantly influence their performance. Examples are the number k for KNN, the minimum impurity decrease for a decision tree or the λ parameter for LASSO. This may even be extended back to the dimensionality reduction techniques, where the final number of dimensions n holds power as well. Parameter variations like this are mostly performed on a search grid, usually combined with cross-validation, which splits up the training data into even more parts, all of which are in turn used as validation set for the ML models that are now trained on a slightly smaller training set. Performing multiple nested cross-validation grid searches, for example to determine which algorithm performs best and with which parameters at the same time,

[4]Note that this grid is being used by a large number of people, so this time estimate may involve waiting periods when these computations are put on hold in favor of higher priority tasks.

[5]Unless of course it is time to go debugging, then those few hours are annoyingly long.

requires keeping track of the different parts of the training set quite meticulously, as in such endeavours it's increasingly easy to mix up which chunks of data should go where.

4.3.2 Different algorithms

Despite the number of machine learning algorithms already included in this thesis, there are still many more possible classifiers and regressors that can be applied to the data. As suggested in section 4.2.1, SGD-based approaches or neural networks might provide an advantage. However, more complex algorithms don't necessarily outperform the simpler ones and are almost always harder to interpret. Looking back at figure 2.18 for example, a neural network could probably learn the boundaries given enough training data, but so can the decision tree, and one of those two is definitely superior in interpretability. Nevertheless, the best suited algorithm is unique for each dataset, so trying new ones is always an option.

4.3.3 Multilabel classification

In this thesis, all ML algorithms have been trained on one feature at a time only. This restriction is not required, as most machine learning algorithms are capable of predicting multiple labels simultaneously for each sample. Not only could the classifiers benefit from this if some features can be predicted more accurately together than alone, but if that is the case, such performance improvement would also implicate a correlation between the features involved. That may in turn help to get a better understanding of the underlying physics, which is after all still the goal.

Chapter 5

Conclusion

This thesis investigated what characteristics of the next galactic supernova can be predicted given a gravitational wave observation. Using a number of selected supernova models, data was simulated for a weeks worth of observations for each model. Stationary, Gaussian noise was added with varying signal to noise ratios. The data was then reconstructed using a wavelet-based gravitational wave search algorithm called BayesWave. Using a range of applicable Python packages, a machine learning environment was set up, where a number of techniques were compared.

To make predictions for the numerical characteristics, mass and rotational velocity, linear regression slightly outperformed the more complex LASSO approach, although there is quite a lot of room for improvement for both of them. For the non-numerical characteristics, the K-nearest neighbour approach had more trouble predicting the explosion mechanism, while for the presence of standing accretion shock instability, prompt convection and rotation the decision tree classifier suffered the lesser performance. The support vector machine classifier, using a one-versus-the-rest approach, generally achieved the best results with an accuracy vs. SNR curve flattening near 100% accuracy above a signal to noise ratio of about 30 to 40 for every predicted characteristic. While there are certainly aspects that can be improved upon or extended, the idea of using machine learning to extract information from gravitational wave observations about the underlying physics of supernovae has certainly been proven to be achievable.

References

Abbott, B. P. et al. (2016). "Observation of gravitational waves from a binary black hole merger". *Physical Review Letters*, 116(6). DOI: 10.1103/physrevlett.116.061102.

Abbott, B. P. et al. (2017). "GW170814: a three-detector observation of gravitational waves from a binary black hole coalescence". *Physical Review Letters*, 119(14). DOI: 10.1103/physrevlett.119.141101.

Abbott, R. et al. (2021). "All-sky search for short gravitational-wave bursts in the third Advanced LIGO and Advanced Virgo run". *Physical Review D*, 104(12). DOI: 10.1103/physrevd.104.122004.

Andresen, H. et al. (2017). "Gravitational wave signals from 3D neutrino hydrodynamics simulations of core-collapse supernovae". *Monthly Notices of the Royal Astronomical Society*, 468(2), 2032–2051. DOI: 10.1093/mnras/stx618.

Andresen, H. et al. (2019). "Gravitational waves from 3D core-collapse supernova models: the impact of moderate progenitor rotation". *Monthly Notices of the Royal Astronomical Society*, 486(2), 2238–2253. DOI: 10.1093/mnras/stz990.

Bethe, H. A. and Wilson, J. R. (1985). "Revival of a stalled supernova shock by neutrino heating". *The Astrophysical Journal*, 295, 14–23. DOI: 10.1086/163343.

Bisnovatyi-Kogan, G. S. (1971). "The explosion of a rotating star as a supernova mechanism." *Soviet Astronomy*, 14, 652.

Blondin, J. M., Mezzacappa, A., and DeMarino, C. (2003). "Stability of standing accretion shocks, with an eye toward core-collapse supernovae". *The Astrophysical Journal*, 584(2), 971–980. DOI: 10.1086/345812.

Bugli, M., Guilet, J., and Obergaulinger, M. (2021). "Three-dimensional core-collapse supernovae with complex magnetic structures – I. Explosion dynamics". *Monthly Notices of the Royal Astronomical Society*, 507(1), 443–454. DOI: 10.1093/mnras/stab2161.

Burrows, A. et al. (2006). "An acoustic mechanism for core-collapse supernova explosions". *New Astronomy Reviews*, 50(7). Astronomy with Radioactivities. V, 487–491. DOI: https://doi.org/10.1016/j.newar.2006.06.046.

Cornish, N. J. and Littenberg, T. B. (2015). "Bayeswave: Bayesian inference for gravitational wave bursts and instrument glitches". *Classical and Quantum Gravity*, 32, 135012, 135012. DOI: 10.1088/0264-9381/32/13/135012. arXiv: 1410.3835 [gr-qc].

Cornish, N. J. et al. (2021). "Bayeswave analysis pipeline in the era of gravitational wave observations". *Phys. Rev. D*, 103(4), 044006. DOI: 10.1103/PhysRevD.103.044006. arXiv: 2011.09494 [gr-qc].

Dálya, G. (2021). "Bevezetés a csillagászatba (Introduction to Astronomy)". ISBN 978-963-8361-58-5. ELKH CSFK. Chap. 5. Stars, 227–301.

Dálya, G., Raffai, P., and Bécsy, B. (2021). "Bayesian reconstruction of gravitational-wave signals from binary black holes with nonzero eccentricities". *Classical and Quantum Gravity*, 38(6), 065002. DOI: 10.1088/1361-6382/abd7bf.

Hamacher, D. W. (2014). "Are supernovae recorded in indigenous astronomical traditions?" DOI: 10.48550/ARXIV.1404.3253.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer.

Hirata, K. et al. (1987). "Observation of a neutrino burst from the supernova SN1987A". *Phys. Rev. Lett.*, 58 (14), 1490–1493. DOI: 10.1103/PhysRevLett.58.1490.

Kuroda, T., Kotake, K., and Takiwaki, T. (2016). "A new gravitational-wave signature from standing accretion shock instability in supernovae". *The Astrophysical Journal*, 829(1), L14. DOI: 10.3847/2041-8205/829/1/l14.

Kuroda, T. et al. (2017). "Correlated signatures of gravitational-wave and neutrino emission in three-dimensional general-relativistic core-collapse supernova simulations". *The Astrophysical Journal*, 851(1), 62. DOI: 10.3847/1538-4357/aa988d.

Littenberg, T. B. and Cornish, N. J. (2015). "Bayesian inference for spectral estimation of gravitational wave detector noise". *Phys. Rev. D*, 91 (8), 084034. DOI: 10.1103/PhysRevD.91.084034.

McInnes, L., Healy, J., and Melville, J. (2018). "UMAP: uniform manifold approximation and projection for dimension reduction". DOI: 10.48550/ARXIV.1802.03426.

Mezzacappa, A. et al. (1996). "Deciphering core collapse supernovae: is convection the key? I. Prompt convection". DOI: 10.48550/ARXIV.ASTRO-PH/9609006.

Mezzacappa, A. et al. (2020). "Gravitational-wave signal of a core-collapse supernova explosion of a 15 solar masses star". *Physical Review D*, 102(2). DOI: 10.1103/physrevd.102.023027.

Migenda, J. (2017). "Astroparticle physics in Hyper-Kamiokande". *PoS*, EPS-HEP2017, 020. DOI: 10.22323/1.314.0020.

O'Connor, E. P. and Couch, S. M. (2018). "Exploring fundamentally three-dimensional phenomena in high-fidelity simulations of core-collapse supernovae". *The Astrophysical Journal*, 865(2), 81. DOI: 10.3847/1538-4357/aadcf7.

Ott, C. D. et al. (2013). "General-relativistic simulations of three-dimensional core-collapse supernovae". *The Astrophysical Journal*, 768(2), 115. DOI: 10.1088/0004-637x/768/2/115.

Pan, K.-C. et al. (2021). "Stellar mass black hole formation and multimessenger signals from three-dimensional rotating core-collapse supernova simulations". *The Astrophysical Journal*, 914(2), 140. DOI: 10.3847/1538-4357/abfb05.

Pedregosa, F. et al. (2011). "Scikit-learn: Machine learning in Python". *Journal of Machine Learning Research*, 12, 2825–2830.

Powell, J., Cerda-Duran, P., and Marek, S. (2019). *Waveforms for O3 CCSN search paper*. URL: https://wiki.ligo.org/Bursts/O3SearchWaveforms.

Powell, J. and Müller, B. (2019). "Gravitational wave emission from 3D explosion models of core-collapse supernovae with low and normal explosion energies". *Monthly Notices of the Royal Astronomical Society*, 487(1), 1178–1190. DOI: 10.1093/mnras/stz1304.

Powell, J. and Müller, B. (2020). "Three-dimensional core-collapse supernova simulations of massive and rotating progenitors". *Monthly Notices of the Royal Astronomical Society*, 494(4), 4665–4675. DOI: 10.1093/mnras/staa1048.

Powell, J., Müller, B., and Heger, A. (2021). "The final core collapse of pulsational pair instability supernovae". *Monthly Notices of the Royal Astronomical Society*, 503(2), 2108–2122. DOI: 10.1093/mnras/stab614.

Radice, D. et al. (2019). "Characterizing the gravitational wave signal from core-collapse supernovae". *The Astrophysical Journal*, 876(1), L9. DOI: 10.3847/2041-8213/ab191a.

Raza, N. et al. (2022). "Prospects for reconstructing the gravitational-wave signals from core-collapse supernovae with Advanced LIGO-Virgo and the Bayeswave algorithm". DOI: 10.48550/arXiv.2203.08960.

Rozwadowska, K., Vissani, F., and Cappellaro, E. (2021). "On the rate of core collapse supernovae in the Milky Way". *New Astronomy*, 83, 101498. DOI: https://doi.org/10.1016/j.newast.2020.101498.

Saeys, Y. (2021). "Machine learning". English. Course notes for the Ghent University class on Machine Learning (C003758A).

Scheidegger, S. et al. (2010). "The influence of model parameters on the prediction of gravitational wave signals from stellar core collapse". *Astronomy and Astrophysics*, 514, A51. DOI: 10.1051/0004-6361/200913220.

Shen, G., Horowitz, C. J., and Teige, S. (2011). "New equation of state for astrophysical simulations". *Physical Review C*, 83(3). DOI: 10.1103/physrevc.83.035802.

Steiner, A. W., Hempel, M., and Fischer, T. (2013). "Core-collapse supernova equations of state based on neutron star observations". *The Astrophysical Journal*, 774(1), 17. DOI: 10.1088/0004-637x/774/1/17.

Szczepańczyk, M. J. (2020). *SN library*. URL: https://git.ligo.org/marek.szczepanczyk/sngw/-/blob/master/Toolbox/sn_toolbox/sn_library.py.

Szczepańczyk, M. J. et al. (2021). "Detecting and reconstructing gravitational waves from the next galactic core-collapse supernova in the advanced detector era". *Physical Review D*, 104(10). DOI: 10.1103/physrevd.104.102002.

Woosley, S. E., Heger, A., and Weaver, T. A. (2002). "The evolution and explosion of massive stars". *Rev. Mod. Phys.*, 74 (4), 1015–1071. DOI: 10.1103/RevModPhys.74.1015.

Woosley, S. E. and Weaver, T. A. (1995). "The evolution and explosion of massive stars. II. Explosive hydrodynamics and nucleosynthesis". *The Astrophysical Journal Supplement*, 101, 181. DOI: 10.1086/192237.

Woosley, S. and Heger, A. (2007). "Nucleosynthesis and remnants in massive stars of solar metallicity". *Physics Reports*, 442(1). The Hans Bethe Centennial Volume 1906-2006, 269–283. DOI: https://doi.org/10.1016/j.physrep.2007.02.009.

Zhao, T. and Lattimer, J. M. (2022). "Universal relations for neutron star f-mode and g-mode oscillations". DOI: 10.48550/ARXIV.2204.03037.

Appendix A

Fysische eigenschappen van supernovae bepalen uit observaties van zwaartekrachtsgolven

Op 24 februari 1987 werd de meest recente supernova in het nabije universum waargenomen. De explosie vond plaats in de Grote Magellaanse wolk, op een afstand van ongeveer $50\ \mathrm{kpc}$. Het speciale aan supernova SN1987A is dat deze niet enkel optisch waargenomen werd. Voor de eerste (en voorlopig ook laatste) keer werden ook neutrino's geobserveerd die afkomstig waren van een supernova. Sinds enkele jaren zijn de mogelijkheden uitgebreid met nogmaals een nieuw waarnemingsgebied: zwaartekrachtsgolven. Computersimulaties tonen aan dat supernovae in staat zijn om zwaartekrachtgolven te produceren. Daarom is de voorbereiding op de volgende supernova binnen de Melkweg nu al begonnnen.

Het doel van deze thesis is te bepalen welke informatie er uit de waarneming van zwaartekrachtgolven kan geëxtraheerd worden. Hierbij ligt de focus zowel op fysische eigenschappen zoals de massa en rotatiesnelheid als op kenmerken van de explosie zoals het explosiemechanisme. Een machine learning algoritme zal getraind en getest worden op gesimuleerde data. Aan de hand van de testdata wordt gemeten hoe nauwkeurig elke eigenschap bepaald kan worden.

A.1 Introductie

Wanneer de nucleaire brandstof in de kern van een ster is opgebrand, stort de ster in onder zijn eigen zwaartekracht. Voor sterren onder de Chandrasekhar-limiet[1] resulteert dit in een witte dwerg, ondersteund door de ontaardingsdruk van de elektronen. Voor zwaardere sterren is verdere instorting onvermijdelijk tot op het punt waar de ontaardingsdruk van de nucleonen voelbaar wordt. Door die plotse extra uitwaartse druk kaatst de kern terug en wordt de instortende materie door een schokgolf naar buiten geduwd. Wanneer de schokgolf het oppervlak van de instortende materie bereikt, wordt een enorme hoeveelheid energie in de vorm van licht het heelal in gestuurd, waarbij de supernova vaak helderder is dan het omringende sterrenstelsel.

[1] ongeveer $1.44\mathrm{M}_{\odot}$

Na de initiële instorting zijn nog weinig details bekend over supernovae. Aangezien licht pas vrijkomt op het einde van de explosie, wanneer de schokgolf het oppervlak bereikt, zijn andere soorten waarnemingen nodig om binnen in de exploderende ster te kijken. Observaties van neutrino's en zwaartekrachtsgolven hebben echter een beperkt bereik, waardoor supernovae buiten het Lokale Universum[2] geen waardevolle detectie kunnen opleveren. Er zal dus gewacht moeten worden tot de volgende ster binnen de Melkweg explodeert, wat helaas slechts één of twee keer per eeuw gebeurt. Daarom is het belangrijk dat uit de volgende waarneming zo veel mogelijk informatie gehaald wordt, zoals bijvoorbeeld de toestandsvergelijking, het explosiemechanisme, en de aanwezigheid van instabiliteiten, convectie en andere oscillaties.

Zoals eerder vermeldt, zijn zwaartekrachtsgolven één van de mogelijke observabelen om binnenin de supernova te kijken. Ze worden veroorzaakt door de veranderende versnelling van grote massa's, onder de voorwaarde dat deze niet sferisch symmetrisch zijn. Supernovae voldoen aan deze voorwaarde, aangezien er bij het optreden van de verschillende effecten hierboven geen sferische symmetrie meer overblijft. Wanneer deze zwaartekrachtsgolven de aarde bereiken, zullen ze waargenomen worden door verschillende zwaartekrachtgolfdetectoren. Vier van deze detectoren worden hier gesimuleerd: de twee detectoren van LIGO in de Verenigde Staten, de Virgo detector in Italië en de KAGRA detector in Japan.

A.2 Methoden

Om te bepalen hoe zwaartekrachtsgolven afkomstig van supernovae eruitzien, zijn reeds vele modellen ontwikkeld. Van deze modellen zijn er 14 geselecteerd die in deze thesis gesimuleerd zullen worden. Bij de selectie zijn de modellen zo gekozen dat een zo groot mogelijk volume in de parameterruimte gevuld wordt. Van elk model zijn vervolgens 1000 zwaartekrachtsgolven gesimuleerd, met willekeurige invalshoeken en signaal-ruisverhoudingen tussen 10 en 100.

De gesimuleerde golven worden vervolgens verwerkt met BayesWave, een programma dat aan de hand van golfpakketten en de regel van Bayes de golven zonder ruis probeert te reconstrueren. Hoewel de originele golven beschikbaar zijn omdat ze hier gesimuleerd zijn, moet gebruik gemaakt worden van de verwerkte golven om zo dicht mogelijk aan te sluiten bij een reële waarneming.

Van de vele uitvoermogelijkheden die BayesWave biedt, is hier het vermogensspectrum gekozen als representatie van de gereconstrueerde golven. Na opsplitsing van de data in training- en testdata wordt eerst een dimensieverlaging uitgevoerd. Twee technieken worden hiervoor getest: PCA (componentanalyse) en UMAP (projectie op basis van lokale topologie). Voor de niet-numerieke kenmerken worden na deze projectie een decision tree, een support vector machine en een nearest neighbour algoritme getest. De numerieke eigenschappen worden bepaald aan de hand van een lineaire regressie en een uitbreiding daarop genaamd LASSO.

[2] enkele honderden kpc

A.3 Resultaten

De dimensieverlagingstechnieken waren succesvol en konden de data met een factor 45 verkleinen. Een algemene trend in de prestaties van de machine learning algoritmes was het dalen van de foutmarge met toenemende signaal-ruisverhouding, hoewel de fout soms onverklaarbaar toch toenam bij de allerhoogste signaal-ruisverhoudingen. Verder waren de voorspellingen voor de niet-numerieke eigenschappen over het algemeen nauwkeuriger dan die voor de numerieke. Of de combinatie van algoritmes met PCA of UMAP beter was, hangt af van de eigenschap waar het over gaat. Opvallend zijn 2 modellen die vaak met elkaar verward werden, dit komt waarschijnlijk omdat ze uit weinig componenten bestaan en daardoor gelijkaardig voorgesteld werden. Het weglaten van één van deze modellen uit de analyse verbeterde de prestaties van de voorspellingsalgoritmes.

A.4 Toekomstig werk

Natuurlijk is er altijd ruimte voor verbetering en kunnen de hier beschreven algoritmes uitgebreid worden. Zo kan er bijvoorbeeld rekening gehouden worden met nieuwe detectoren of kunnen er nieuwe supernovamodellen toegevoegd worden. Voor sommige van deze uitbreidingen zal een deel van de analyse opnieuw moeten plaatsvinden. Ook een mogelijkheid is het zoeken naar aanpassingen aan de gebruikte algoritmes die incrementeel werken vergemakkelijken.

Verder kan het de moeite lonen om eens dieper in te gaan op de dimensieverlaging. Zo is het niet ondenkbaar dat er informatie verscholen zit in de representatie van de data na dimensieverlaging. Ook hebben sommige van de machine learning algoritmes een afstelbare parameter die de prestaties kan beïnvloeden en waarvoor de optimale waarde kan gezocht worden.

Ten slotte is het ook mogelijk om compleet nieuwe algoritmes te testen. De gebruikte algoritmes zijn goed interpreteerbaar, maar complexere systemen als neurale netwerken kunnen ook het proberen waard zijn. Een uitbreiding aan de algoritmes kan ook gebeuren door meerdere kenmerken tegelijk te proberen voorspellen. Als de nauwkeurigheid daardoor stijgt, kan daaruit iets bijgeleerd worden over de fysica achter deze kenmerken en hun correlatie.

A.5 Conclusie

In deze thesis werd onderzocht welke eigenschappen van een supernova bepaald kunnen worden aan de hand van de waarneming van zwaartekrachtsgolven. Na het simuleren van een selectief aantal modellen, inclusief ruis, werd een poging gedaan om hun eigenschappen te voorspellen op basis van de gereconstrueerde data. Enkele machine learning algoritmes werden hierbij vergeleken met elkaar. Hoewel er zeker ruimte is voor verbetering en uitbreiding, is het basisidee om machine learning te gebruiken om de onderliggende fysica te onderzoeken wel werkbaar gebleken.

Appendix B

Comments on source code

The aim of this appendix is to take a more detailed look at the source code used to create the results. This may give a better insight into why some choices were made or what obstacles were overcome to get to the final version. All source code was written in Python 3, except for some necessary control commands or scripts, which were implemented in bash.

There are some general considerations about possible improvements that apply to all code presented. As a first remark, a lot of the code still contains hardcoded file paths. While that was no problem for the analysis here, it may be cumbersome to change them all during further research. This may incentivize a revision of the code to make it more accessible.
An even more courageous undertaking would be an attempt to standardize a data format for GW data, so there would no longer be a need for model-specific processing.

A second path that may prove beneficial involves rewriting the existing functions into command line programs or building a class structure around them, which would provide even more modularity and make it easier to redo a certain part of the analysis without the need to rewrite the subsequent parts.

B.1 SN toolbox

The appendix file `sn_library.py` contains the code described in this section. This is a shortened version of the original SN toolbox (Szczepańczyk, 2020). It uses `numpy` and `scipy` to implement some often used data transformations. The only small modification here is the addition of the smoothing parameter s in the `sn_resample_wave` call signature to be set upon function call rather than to use a default value of zero. This is used in cases where the default behaviour leads to the presence of NaNs.

B.2 Generating the waveforms

The appendix file `create_gwf.py` contains the code described in this section. After loading in some GW specific packages, the library functions from section B.1 are included through an `exec` function call. They could not be imported in the usual way, since the `sn_library.py` file resided in another folder on the computer grid.

The next few lines after that set up the model specific configuration. Datafiles and output folders are identified and created and the length and number of samples are set[1].

Next up are a few helper functions. `so3_matrix` generates a random SO(3) matrix that is necessary for some models to give a random orientation to the general waveforms that are loaded in by `ampl`. Because the data is not available in a standard format, the function `ampl` needs to be finetuned for each model, for example to use an SO(3) matrix if the data is delivered as quadrupole contributions instead of the usual h_+ and $h_\times$ polarization data. Data encoded in milliseconds instead of seconds also belongs to the possibilities. The second pair of functions cooperate to form a high pass filter.

After that, the same code is run 1000 times to generate the 1000 samples needed for each model. Resampling is important to make sure all timestamps are evenly spaced, as the remaining steps depend on that consistency. The equidistantly sampled data is then ready to be projected onto the detectors. What follows is a long list of operations aimed at the padding of the data into exactly $10\ \mathrm{s}$ long samples, ending in a conversion to an appropriate data format, which is then saved in the preconfigured file location. Finally, the randomly generated parameters are also saved in order to later confirm their true randomness.

B.3 Preparation and setup for BayesWave runs

B.3.1 Rescaling the waveforms

The appendix file `rescale_definitions.py` contains the code described in this section. It encompasses all code required to rescale the generated waveforms to a new, randomly distributed set of signal to noise ratios. After importing the required packages, where necessary through another `exec` call, all other functionality is gathered in functions. The function `get_wave_filenames` is pretty straightforward and mainly keeps the long list comprehension out of the way in further uses. The next function, `get_channels_fasds_asds`, gathers the sensitivity curves for the four detectors to be used in the SNR calculations. With `sn_library_snr`, the total SNR for one waveform is then determined as the square root of the sum of squared SNRs of the individual detector components. It is then up to the `rescaler` to combine these calculated values with the new ones generated by `get_random_snrs` and refactor the data through a multiplication of the ratio of old and new SNRs. During the setup of this code, a comparison was made between the calculated values with `sn_library_snr`

[1] As indicated before, 1000 samples of length $10\ \mathrm{s}$ with an intersample distance of $600\ \mathrm{s}$

and those calculated by BayesWave later on. This produced a discrepancy factor of approximately 1.068, which has been added to the code. It is however not yet understood where this discrepancy originates. A comparison between the source codes might be a valid next step in investigating this. The functions described above are combined in `rescale_auto`, which finishes with a call to `write_snr_info` to gather the new and old SNRs in a data file, which may prove convenient later on.

B.3.2 Preparing the configuration files

The appendix file `preparation_definitions.py` contains the code described in this section. The structure is similar to `rescale_definitions.py` as the final function, `prepare_auto`, gathers the others in one call. This creates the auxiliary files that BayesWave uses to find the data and set up its own configuration. It even executes one of the scripts it generated earlier, after which the start of the BayesWave reconstruction process is only one prewritten command away.

B.3.3 Combining and executing

The appendix file `prepare.py` contains the code described in this section. This script does not much more than combining the functions `rescale_auto` and `prepare_auto` from `rescale_definitions.py` and `preparation_definitions.py` respectively. If a different setup would be preferred, which wouldn't use the default options of all the intermediate functions, this could also be done here. The only thing that does need to be set here is the correct path, so the functions will perform their actions in the right place.

B.4 Machine learning

B.4.1 Supporting functions

The appendix file `ml_defs.py` contains the code described in this section. It contains code that is often reused during the gathering and processing of the data for the machine learning algorithms.

The first two functions serve solely to extract the data from the BayesWave output. As that output is always structured the same way, the relevant subpaths can be hardcoded without causing trouble across different models. Because opening and reading files is a slow process, a multiprocessing pool is used to speed this up by handling multiple files in parallel.

The two functions succeeding these first two are in charge of handling the data from that point on. Once the data has been extracted from Bayeswave output, it is saved into a few centralized files, allowing for a faster readout since less files need to be opened[2].

[2] a few minutes as compared to a few hours

Next up is a class built to gather information on the parameter values for each model. It also aids in retrieving that data efficiently during the training and testing process.

With `acc_snr_plots`, the data gathered during the ML phase is transformed into error or accuracy vs. SNR plots. This function mainly takes care of presenting the data correctly and adjusting the plot to maximize readability.

The last function, `sn_gw_ml_alg`, does the heavy lifting during the ML phase. It takes a certain dataset, already split into training and test data, and trains the accompanying ML algorithm on this data. It then continues with a prediction of the test data, followed by scoring these predictions per SNR bin through `get_snr_scores`, which is defined just before it and applies the correct performance measure depending on the feature being numerical or not.

B.4.2 Execution

The appendix file `ml.py` contains the code described in this section. The first 60 lines approximately are concerned with importing the correct machine learning models, as well as defining the data locations and model parameters. If the data has not been extracted yet, this is then the next step, otherwise this has already been completed before and this step can be skipped.

The main part of this program is the quadruply nested for loop that follows. The outer loop ensures that the whole analysis is performed for both dimensionality reduction techniques. Two score lists are defined, one for the numerical features and one for the non-numerical. Each of the combinations of a dimensionality reduction technique with a machine learning algorithm is then trained ten times, each time saving the result, which in the end is plotted using the function described in the previous section.

A few choices were made regarding the models used, which can be described here alongside their programmatic definition. It is for these values that section 4.3.1 prescribes a future grid search to finetune them. The principal component analysis is set to retain an appropriate amount of data to cover 98% of the variance in the original data. As mentioned before, this reduces the number of components to approximately 730. The uniform manifold approximation and projection does not have the option to cover a percentage, it needs an integer number of components to retain. To this end, the average number of components from the PCA coupled algorithms is used for UMAP. This has as a benefit that both techniques have the same amount of data available, meaning the results can be used to figure out which is the better representation. The train-test split is a $90\% - 10\%$ split of the data. This split is stratified using not only the model a sample belongs to, but also the SNR bin it is a part of. This ensures that all models and all SNR ranges within those models are represented in a similar ratio in the training and in the test set. A choice was made to discard the original data after performing the dimensionality reduction, as keeping it in memory proved to cause a larger slowdown than reading it in for every loop.

The machine learning algorithms themselves also come with some adjustable parameters, where a choice was made here based on intuition. The decision tree classifier has been set to use a minimum impurity decrease of 0.01, as opposed to the default value of 0, which would produce the complete tree and have a large chance of overfitting the trained model. It has also been configured to use a balanced error measure, to account for any remaining imbalance in the classes put into the algorithm. This last statement is also true for the SVM. Additionally, the SVM has been set to fit the intercept, as the normalization of the data is not necessarily maintained by the dimensionality reduction. The KNN algorithm has been set to use 3 times as much neighbours as there are classes to be predicted. The predictions have also been set to take into account the distance between a sample and one of its nearest neighbours, as opposed to treating each neighbour equally, regardless of its distance. This weighting factor also prevents draws, rendering the choice of an odd number of neighbours as mentioned in section 2.4.3 unnecessary. For both numerical algorithms, nothing more than a fit of the intercept was set up, all other parameters, if present, keep their default value.

Running this program takes quite a while. Depending on whether or not the data extraction has already been completed, a few hours can be spared there already. Nevertheless, the PCA reduction on the full dataset can take over an hour each time. The UMAP reduction performs better in terms of timing and only needs about 15 minutes to get the job done. After that, the ML algorithms can take anywhere between a few seconds to a few minutes to train and test, with the largest time consumer being the SVM.